

Deep Lineage: Single-cell lineage tracing and fate inference using deep learning

Highlights

- Deep Lineage jointly models scRNA-seq and clonal lineage across developmental time
- It predicts early cell fate bias and imputes unmeasured single-cell time points
- Bidirectional RNNs support both forward prediction and backward state imputation
- SHAP-guided *in silico* perturbations propose testable fate-control hypotheses

Authors

Mehrshad Sadria, Allen Zhang,
Haizhi Gary Lai, Gary D. Bader

Correspondence

msadria@uwaterloo.ca (M.S.),
gary.bader@utoronto.ca (G.D.B.)

In brief

Sadria et al. present Deep Lineage, a deep-learning method that combines time-series single-cell RNA sequencing with clonal lineage tracing to predict early cell fate bias and reconstruct unmeasured expression states. Benchmarked against a fate-aware simulator and existing methods, it recovers previously reported fate-associated genes and generates testable *in silico* perturbation hypotheses.

Methods

Deep Lineage: Single-cell lineage tracing and fate inference using deep learning

Mehrshad Sadria,^{1,2,9,*} Allen Zhang,¹ Haizhi Gary Lai,^{2,3} and Gary D. Bader^{2,3,4,5,6,7,8,*}

¹Department of Applied Mathematics, University of Waterloo, Waterloo, ON, Canada

²The Donnelly Centre, University of Toronto, Toronto, ON, Canada

³Factored AI, Palo Alto, CA, USA

⁴Department of Molecular Genetics, University of Toronto, Toronto, ON, Canada

⁵Department of Computer Science, University of Toronto, Toronto, ON, Canada

⁶The Lunenfeld-Tanenbaum Research Institute, Sinai Health System, Toronto, ON, Canada

⁷Princess Margaret Cancer Centre, University Health Network, Toronto, ON, Canada

⁸CIFAR Multiscale Human Program, CIFAR, Toronto, ON, Canada

⁹Lead contact

*Correspondence: msadria@uwaterloo.ca (M.S.), gary.bader@utoronto.ca (G.D.B.)

<https://doi.org/10.1016/j.cels.2026.101759>

SUMMARY

Recent advances in single-cell RNA sequencing and lineage tracing techniques have provided valuable insights into the temporal changes in gene expression during development, tumor progression, and disease onset. However, there are few computational methods available to analyze this information to help understand multicellular dynamics. We introduce Deep Lineage, a deep-learning method for analyzing time-series single-cell RNA sequencing with matched lineage tracing data. Our method accurately predicts early cell fate biases and gene expression profiles at different time points within a clone, surpassing existing methods in fate prediction accuracy. We further validate our model using LineageSim, a fate-aware lineage simulator, demonstrating robustness to technical noise such as barcode dropout and clone misidentification. Additionally, we apply Deep Lineage to *in silico* perturbations in cellular reprogramming and hematopoiesis to identify key genes and pathways associated with cell fate determination and propose testable perturbation hypotheses for experimental follow-up.

INTRODUCTION

Developmental biology aims to understand the processes by which single cells divide and differentiate into diverse cell types in different tissues.¹ Single-cell RNA sequencing (scRNA-seq) is a powerful tool that can investigate developmental landscapes by profiling gene expression in individual cells at different stages.² While scRNA-seq is valuable for investigating genetic mechanisms in cell differentiation, its destructive nature restricts it to capturing single-time-point snapshots of a developing system. This constraint poses challenges in tracing lineage relationships and tracking cell trajectories over time.³

To overcome this limitation, a range of methods have been developed to study temporal relationships between cells. One class of computational methods, trajectory inference, predicts the order of cells along a developmental path based on the idea that cells that are developmentally related will exhibit similar gene expression profiles.⁴ However, trajectory inference methods are limited in their capacity to capture the complexity of developmental processes, such as lineage branching and cellular plasticity, and can be negatively affected by technical factors and noisy data.^{5,6} Recent experimental lineage tracing

technologies that use CRISPR-Cas9 genome editing to record cell division events provide another avenue for studying cell development.⁷ Using data produced by these technologies, computational methods have been developed to infer developmental trajectories by considering both cell lineage and cell state information from multiple time points.^{8–11} Integration of these two types of information has made predictions more accurate.¹² However, despite these advances, accurate prospective cell fate prediction remains challenging.¹³

Here, we present Deep Lineage, a computational model that integrates lineage tracing and scRNA-seq data to learn cellular trajectories, enabling the prediction of gene expression and cell type at any given time point along a trajectory. Deep Lineage employs recurrent neural networks, specifically long short-term memory (LSTM), bidirectional long short-term memory (Bi-LSTM), or gated recurrent units (GRUs), to model complex sequential dependencies and temporal dynamics of a cellular trajectory. An autoencoder-learned embedding captures essential features of the data to simplify input to the LSTM. We employed LineageSim,¹⁴ a fate-aware lineage simulator, to benchmark fate prediction against a known ground truth, in addition to analyzing real-world scRNA-seq data containing lineage tracing

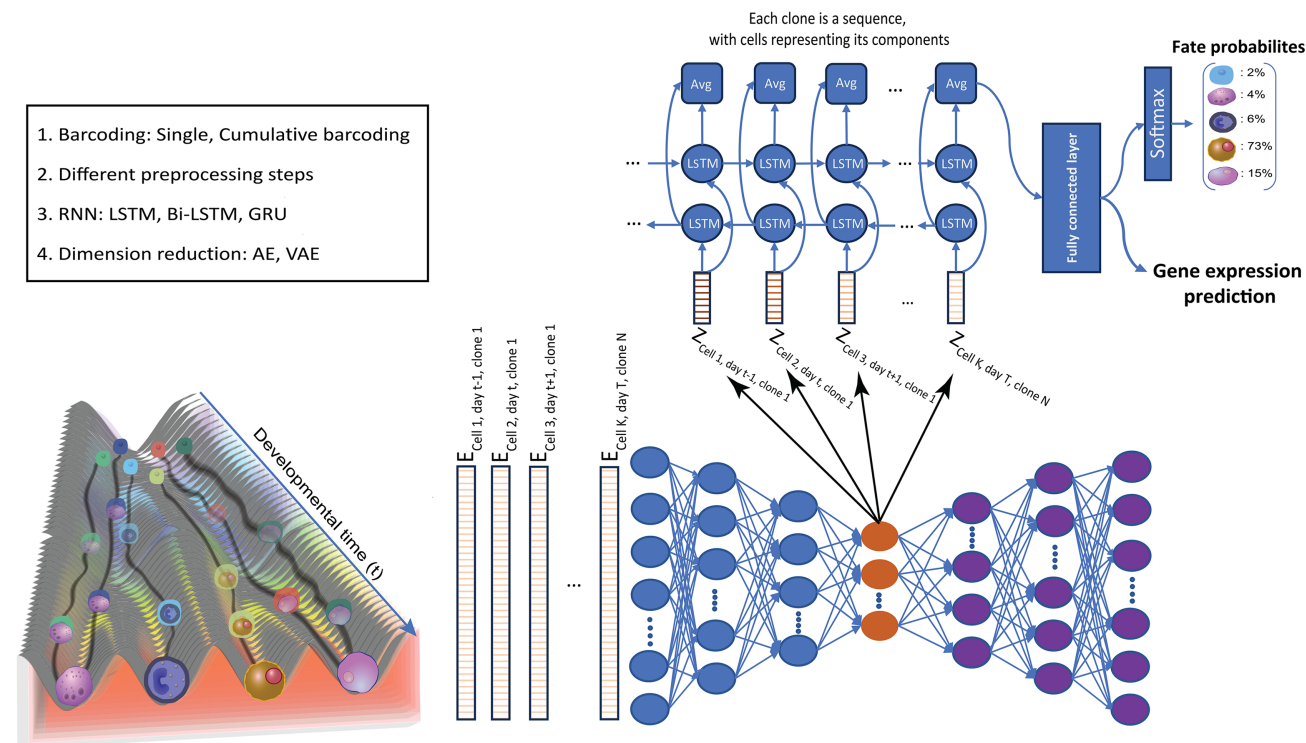


Figure 1. Predicting gene expression and early cell fate bias via combined scRNA-seq and lineage tracing

A visual depiction of Deep Lineage is presented. On the left, the Waddington landscape illustrates the developmental trajectory, showing how cells within each clone (represented by different colors) differentiate into distinct mature cell types originating from stem cells. On the right, by combining scRNA-seq data and lineage tracing information, the gene expression profiles of cells within each clone, as well as the time point, are used as input for the autoencoder. The resulting latent embeddings of cells in a clone are subsequently used as inputs for the recurrent network such as LSTM, Bi-LSTM, or the GRU model. This integrated architecture enables the accurate prediction of both early cell fate bias and gene expression profile of unseen days within a clone. Notably, Deep Lineage offers the flexibility to adapt different preprocessing steps, diverse barcoding techniques, various dimensionality reduction methods, and various deep-learning models (summarized in the box at the top left of this panel, which lists the interchangeable options available at each stage: preprocessing, barcoding strategy, dimensionality reduction method, and recurrent architecture). Abbreviations: scRNA-seq, single-cell RNA sequencing; LSTM, long short-term memory; Bi-LSTM, bidirectional LSTM; GRU, gated recurrent unit. This panel is a schematic illustration and does not display experimental data.

See also Figure S1.

information. Our results demonstrate that Deep Lineage surpasses existing methods in detecting early fate bias and maintains robustness even under conditions of high technical noise, such as lineage barcode dropout. Furthermore, we show Deep Lineage's ability to identify important genes and pathways involved in cell fate decision-making during a developmental process and to generate testable *in silico* perturbation hypotheses describing how candidate interventions could shift gene expression of cell progenies. Such insights can aid in designing experiments to reprogram cells into specific cell types and potentially translate to clinical applications.

RESULTS

Model overview

Deep Lineage uses lineage tracing and multi-time-point scRNA-seq data (at least three time points) to learn a robust model of a cellular trajectory such that gene expression and cell type information at different time points within that trajectory can be predicted. It can be used to predict single-cell time points that have not been measured within a clone, including predicting

cell fate at future time points from early time points (early cell fate bias). An autoencoder-learned embedding captures essential data features and a recurrent network (LSTM, Bi-LSTM,¹⁵ or GRU¹⁶) is used to support cell type classification and gene expression regression tasks (Figure 1). Deep Lineage treats cells and their progenies within a clone as interconnected entities. We conceptualize cellular relationships through “clones,” which represent cells organized within a common lineage. This clone information captures how cells are related over time.

We train the model, including an extensive hyperparameter search and data preprocessing to optimize the model's performance (lowest loss for reconstruction and classification), on predicting held-out time points. We then train the optimized model with all data. To use the model, single-cell gene expression profiles from one or more time points are encoded in the learned autoencoder's low-dimensional representation. We feed this encoding, considering the formation and relationships from clonal information (sequential input encoding), into the recurrent network (LSTM, Bi-LSTM, or GRU) to predict cell fate bias by analyzing cell types within each clone at a predicted future time point (Figure 1). The model also predicts gene expression

profiles for cells at different stages of the trajectory that can be used to impute missing time points. Deep Lineage can incorporate lineage tracing data using either single barcoding or cumulative barcoding methods (Figure S1). Our analysis suggests that lineage tracing with cumulative barcoding can substantially boost the accuracy of the model compared to single barcoding, resulting in more accurate predictions for both early cell fate bias and gene expression patterns at different stages of development. By applying the Shapley additive explanations (SHAP) method¹⁷ to the trained model, Deep Lineage can attribute model predictions to specific input genes, prioritizing candidate genes and regulatory mechanisms linked to different fate outcomes at different developmental stages for downstream investigation.

Deep Lineage accurately predicts gene expression and early cell fate bias in simulated data

To evaluate whether Deep Lineage can accurately predict future gene expression and cell types from early progenitor cells, we used LineageSim,¹⁴ a cell lineage simulator that enables systematic benchmarking of fate inference methods against a known ground truth. We generate a cell lineage tree and simulate transcriptome profiles for 2,000 genes, from which we sample 4,106 cell lineage trajectories comprising 8,192 cells. The simulated data contain two possible terminal cell fates. Each trajectory contains three temporally ordered cells from a lineage: an early progenitor (t_0), an intermediate cell (t_1), and a terminal cell (t_2).

Following the Deep Lineage workflow, we first train autoencoders to embed single-cell gene expression into low-dimensional latent spaces, selecting hyperparameters via grid search (Table S1). We then train a series of Bi-LSTM models for cell fate prediction (classification) and gene expression inference (regression). For cell fate prediction, we train classifiers to predict the fate bias using (1) t_0 embeddings alone, (2) t_0 and t_1 embeddings, or (3) t_0 , t_1 , and t_2 embeddings as an asymptotic performance limit. The classification task is nearly balanced (2,051 versus 2,055 trajectories). In parallel, for gene expression inference, we train regressors to reconstruct the gene expression of held-out time points, including future ($t_0 + t_1 \rightarrow t_2$) and intermediate ($t_0 + t_2 \rightarrow t_1$) predictions.

When trained to predict terminal cell fate using only t_0 embeddings, the classifier achieves accuracy (69.1%) well above chance (50% for a balanced binary task), demonstrating that the model learns fate signals in early progenitor gene expression (Figure 2A). Incorporating t_1 embeddings increases test accuracy to 85.2%, showing additional temporal context refines cell fate prediction. As an upper bound, providing embeddings from all three time points ($t_0 + t_1 + t_2$) results in near-perfect accuracy (99.8%), as expected since the terminal gene expression is included in the input (Figure 2A).

For gene expression prediction, Deep Lineage accurately reconstructs held-out time points within a lineage (Figure 2B). In the future prediction task ($t_0 + t_1 \rightarrow t_2$), predicted expression shows strong agreement with ground truth (Pearson correlation $r = 0.86$). In the intermediate prediction task ($t_0 + t_2 \rightarrow t_1$), the model similarly recovers intermediate states with high fidelity ($r = 0.80$). Per-gene correlations show that most genes (85.9%) have excellent ($r > 0.7$) or good ($0.5 < r \leq 0.7$) agreement between predicted and true expression, while only 2.9% of genes show

poor agreement ($r \leq 0.3$) (Figure 2C). Collectively, these results show that Deep Lineage achieves strong performance in both early fate prediction and gene expression inference in simulated lineage data.

To evaluate the robustness of Deep Lineage under realistic lineage readout conditions, we simulate barcode dropout, where only a fraction of lineage barcodes is detectable due to technical noise. We fix the test trajectories for fair comparison and, at each dropout rate (25%, 50%, and 75%), randomly mark a proportion of cells in the training and validation trajectories as unidentifiable, retaining only trajectories with identifiable cells for model training. We then train Deep Lineage classifiers and regressors using the same workflow as above.

For cell fate prediction, classification accuracy decreases monotonically with increasing barcode dropout (Figure 2D). Performance degrades gradually at intermediate dropout levels: even at 50% barcode dropout, the t_0 only classifier achieves 63.5% accuracy and the $t_0 + t_1$ classifier achieves 78.3%, both well above the random baseline. At 75% dropout, these models fall to chance-level performance, consistent with the severe reduction in available training trajectories. As expected, the $t_0 + t_1 + t_2$ model (effectively a terminal cell type classifier) still maintains high accuracy even under major dropout (Figure 2D).

For gene expression prediction, performance also decreases progressively with increasing barcode dropout (Figure 2E). At 50% dropout, both future ($t_0 + t_1 \rightarrow t_2$) and intermediate ($t_0 + t_2 \rightarrow t_1$) predictions retain high Pearson correlations (for future $r = 0.82$; intermediate $r = 0.71$). At 75% dropout, performance declines sharply for both regression tasks. Across all dropout levels, future prediction is consistently more robust than intermediate, likely reflecting the stronger temporal constraint imposed by observing earlier time points. Overall, Deep Lineage preserves strong cell fate and gene expression inference under moderate levels of barcode dropout.

In addition, we assess Deep Lineage's robustness to clone misidentification by randomly shuffling the terminal cell type labels for a percentage of training and validation trajectories (Figure 2F). As expected, classification accuracy decreases monotonically as the misidentification rate increases. At 25% misidentification, accuracy remains high: relative to the noiseless baseline, performance decreases by 3.57% (t_0), 3.19% ($t_0 + t_1$), and 1.23% ($t_0 + t_1 + t_2$). Models with greater temporal context (i.e., more observed time points) exhibit a more gradual degradation. At 50% misidentification, accuracy drops sharply across all models. At 75% misidentification, the performance ranking reverses, and the $t_0 + t_1 + t_2$ model performs substantially below chance, indicating systematic label inversion at high misidentification rates.

Overall, Deep Lineage demonstrates robustness against moderate noise in barcode dropout and clone misidentification, enabling accurate fate prediction and gene expression inference in simulated lineages, with graceful degradation in performance as noise increases.

Deep Lineage accurately predicts gene expression and early cell fate bias in mouse hematopoiesis

We applied Deep Lineage to the published lineage tracing data of mouse hematopoiesis, which consists of gene expression profiles of 130,887 cells involving 25,289 genes.¹⁸ These clones

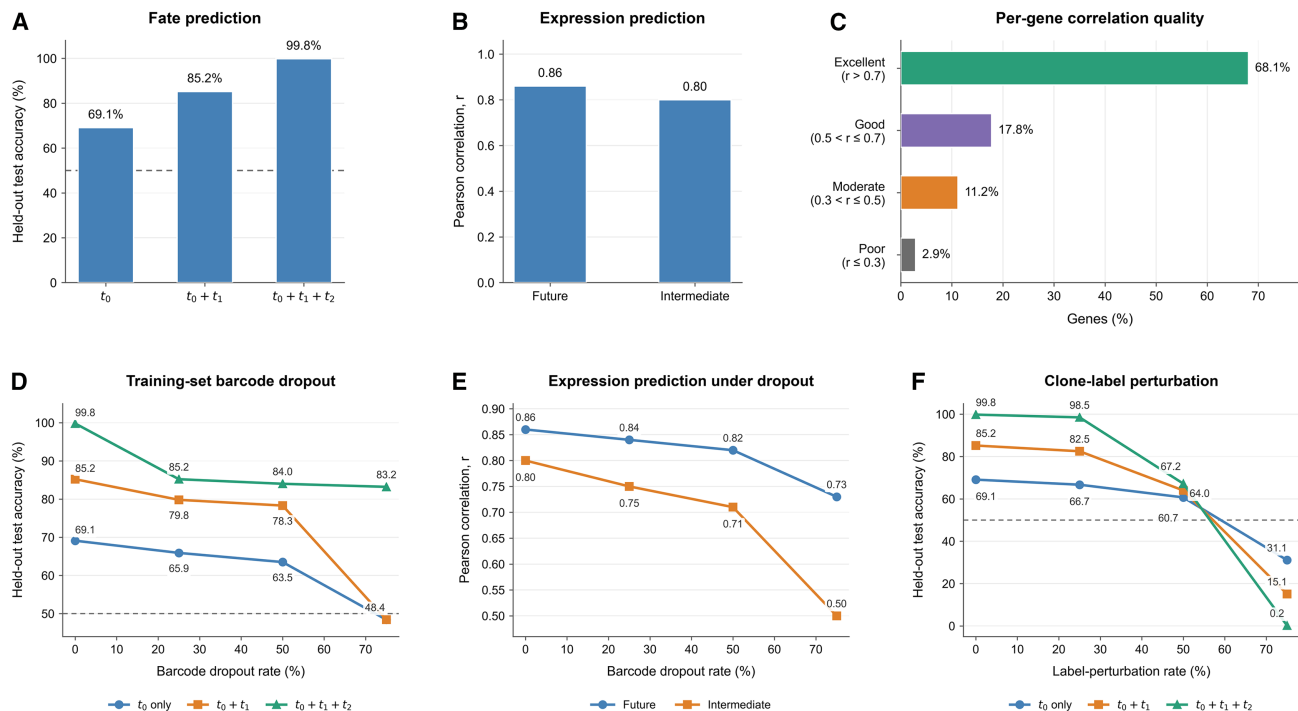


Figure 2. Deep Lineage accurately predicts fate bias and reconstructs gene expression in simulated lineage data and is robust to barcode dropout and clone misidentification

(A) Cell fate prediction accuracy for models using (1) t_0 only, (2) $t_0 + t_1$, or (3) $t_0 + t_1 + t_2$ as input. The dashed line represents the chance level for a balanced binary classification task. Deep Lineage achieves accuracy well above chance from early progenitors (t_0) and improves with additional temporal context, reaching near-perfect performance when all time points are provided.

(B) Time-point gene expression inference performance for future ($t_0 + t_1 \rightarrow t_2$) and intermediate prediction ($t_0 + t_2 \rightarrow t_1$), reported as Pearson correlation between predicted and ground-truth gene expression.

(C) Distribution of per-gene Pearson correlations for gene expression inference, pooled across future and intermediate prediction. Genes are stratified by correlation thresholds (excellent $r > 0.7$; good $0.5 < r \leq 0.7$; moderate $0.3 < r \leq 0.5$; and poor $r \leq 0.3$), indicating that most genes are reconstructed with high accuracy.

(D) Fate prediction accuracy versus barcode dropout rate for models trained with t_0 only, $t_0 + t_1$, or $t_0 + t_1 + t_2$ inputs. Barcode dropout is simulated by marking a fraction of cells in training and validation trajectories as unidentifiable and retaining only trajectories with identifiable cells, while keeping the test set fixed for fair comparison. Accuracy decreases monotonically as dropout increases but remains above random for moderate dropout; high dropout leads to random-level performance.

(E) Gene expression inference performance versus dropout rate for future ($t_0 + t_1 \rightarrow t_2$) and intermediate ($t_0 + t_2 \rightarrow t_1$) models. Future prediction is more robust than intermediate prediction, and both degrade with increasing dropout.

(F) Fate prediction accuracy versus clone misidentification rate. We randomly shuffle terminal fate labels for a fraction of training and validation trajectories to simulate clone misidentification while keeping the test set fixed. Classification accuracy decreases monotonically with increasing misidentification for all input configurations (t_0 , $t_0 + t_1$, and $t_0 + t_1 + t_2$), with a sharp collapse once mislabeled trajectories dominate the training set.

Dataset: LineageSim simulated data ($n = 4,106$ trajectories, 8,192 cells, 2,000 genes; two balanced terminal fates, 2,051 versus 2,055 trajectories; trajectory-level 80%/10%/10% train/validation/test split). Accuracy is evaluated on held-out test trajectories against the known ground-truth fate (chance = 50%); gene expression agreement is quantified by the Pearson correlation coefficient (r). t_0 , t_1 , and t_2 denote the early progenitor, intermediate, and terminal cell of a trajectory, respectively. In (A)–(F), each plotted value is a single summary statistic computed on the held-out test trajectories. In (C), per-gene correlations are grouped into bins; the pooled correlation for each task is reported in (B). In (D)–(F), test trajectories are held fixed across all noise conditions; dropout and label-shuffling are applied only to the training and validation folds. In (F), the panel axis label “label-perturbation rate” is the clone misidentification rate, that is the fraction of training and validation trajectories whose terminal fate labels were shuffled.

See also Table S1.

were sampled at three time points, on days 2, 4, and 6. We focused on the main cell lineage in this dataset, specifically neutrophils, monocytes, and their progenitors (Figure 3A). We trained models to predict the gene expression profile of held-out cells at two time points within a clone predict day 4 (intermediate prediction) by training with days 2 and 6 and predict day 6 (future prediction) by training with days 2 and 4 (Figure 3B), including hyperparameter optimization (see STAR Methods). We calculated the correlation between the gene expression pro-

file of the predicted and held-out cells. The predictions showed strong agreement with held-out observations ($R = 0.81$ for both experiments), indicating that Deep Lineage accurately predicts the gene expression of cells unseen by the model on both day 4 (trained on days 2 and 6) and day 6 (trained on days 2 and 4) (Figures 3C and S2A). The correlation between predicted gene expression and randomly sampled cells was lower, showing the specificity and accuracy of the model’s predictions for the targeted cell types (Figure 3C). We also observed a strong

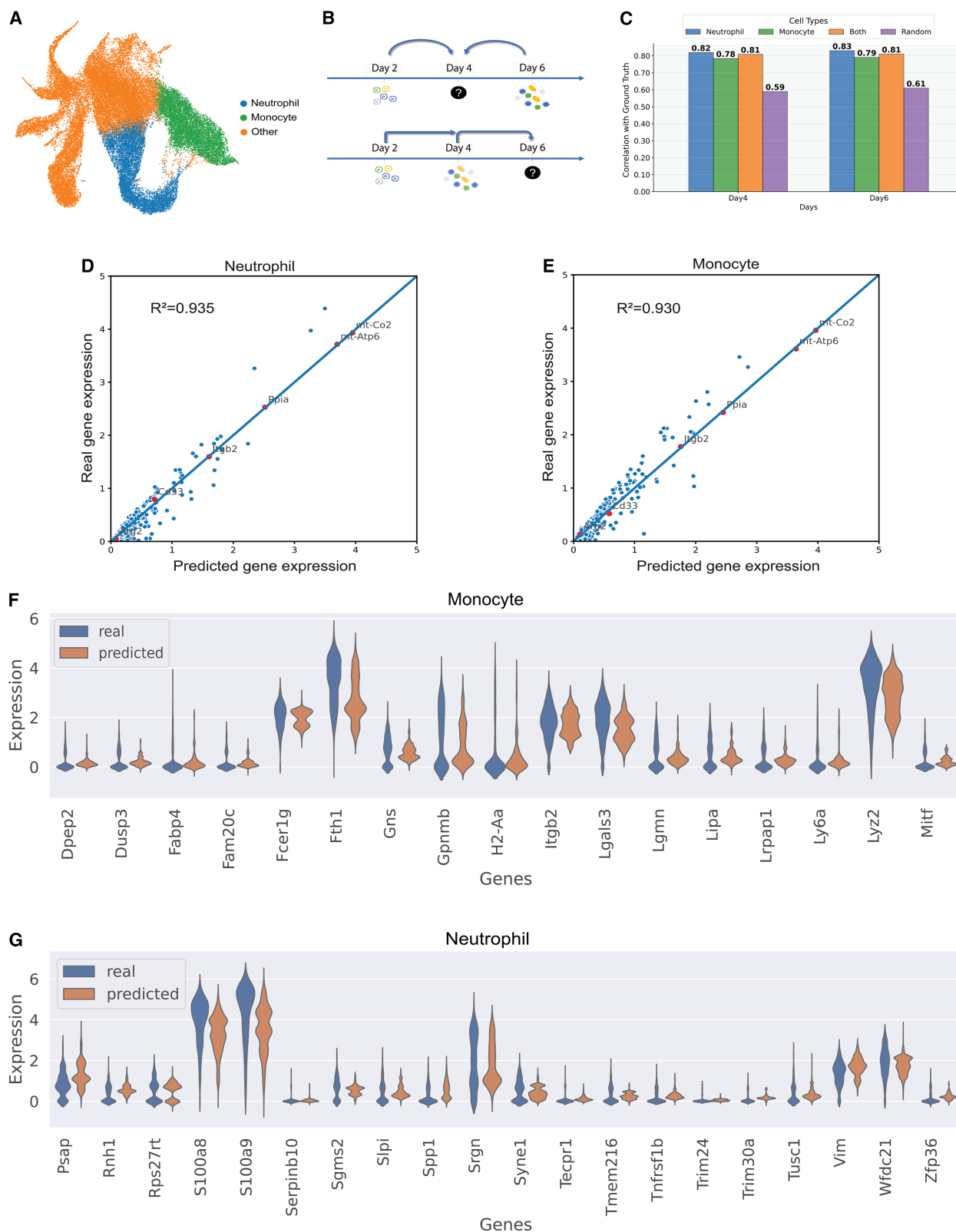


Figure 3. Deep Lineage accurately predicts single-cell gene expression of unseen cells in hematopoiesis

(A) UMAP plot of the hematopoiesis data from Weinreb et al.¹⁸ Each point represents an individual cell, color-coded by cell type.

(B) Schematic representation of the gene expression prediction process using two regression models: one for predicting day 4 (intermediate) gene expression (trained on data from days 2 and 6), and the other for predicting day 6 (future) gene expression (trained on data from days 2 and 4) within a clone.

(legend continued on next page)

positive correlation between the average predicted expression of all genes and actual expression in both monocytes and neutrophils on day 4 (Figures 3D and 3E). Deep Lineage accurately predicts average gene expression and the full distribution of diverse genes for cells within a clone that were previously unseen by the model (Figures 3F and 3G).

To investigate early cell fate bias (Figure 4A) in hematopoiesis, we trained a Bi-LSTM model using the cell type classification task (80% of clones across time points used for training, 10% used for validation, and 10% for testing; see STAR Methods) and used it to predict late time-point (day 6) clonal fate (cell type) from an early time point (day 2 or days 2 and 4). Using only the gene expression of cells on day 2 of a clone, with the masking of data from days 4 and 6, the model attained a predictive accuracy of 76.8% in determining cell fate on day 6 (Figure 4B). Providing gene expression of cells on days 2 and 4 of a clone improved performance slightly, and the model predicted the clonal fate on day 6 with 78.5% accuracy. As a sanity check, to ensure that the model achieves high performance when given data from all three time points, we provided as input the cells within a clone from all time points (10% held-out test clones) and evaluated predicted cell fate of these cells. As expected, the model could almost perfectly predict the fate of the clones (99.2%) (Figure 4B). In essence, the latter is a form of cell type classification (upper bound), as gene expression from the last time point is provided to the model. We also calculated the area under the curve (AUC) for cell fate prediction of this model by using gene expression of cells on days 2 and 4 and all 3 days (Figure 4C). Thus, by using the trained classifier, we predicted the early progenitor bias from an early time point, accurately differentiating between monocyte and neutrophil outcomes. Model performance improved when trained with additional time points. Furthermore, to investigate the importance of lineage tracing and clonal information, we conducted an ablation study in which we randomly sampled cells at each time point, deliberately ignoring clonal data. For the experiment where we trained on day 2 and predicted outcomes for day 6, we constructed trajectories by focusing solely on the day of the cell and its final fate outcome while disregarding the clonal information. The results showed that neglecting lineage tracing information substantially reduces Deep Lineage's prediction accuracy from 76.8% to 52%. Adjusting the model's architecture and hyperparameters resulted in minimal change in accuracy (from 50% to 52%), highlighting the critical role of clonal structure in fate prediction (Figure S3A; Table S2). Next, we assessed our method alongside established models: CoSpar,⁹ Super-OT,¹⁰ Waddington-OT (WOT),⁸ and generative adversarial network (GAN)-based optimal transport (OT),¹⁹ in predicting the fate of held-out clones during training. Our results indicate that Deep

Lineage outperforms all other existing methods, demonstrating higher accuracy in predicting early fate bias (Figure 4D). Specifically, the comparative analysis revealed that Deep Lineage achieved an accuracy of 76.8%. In contrast, other methods such as CoSpar attained a rate of 65%, Super-OT exhibited a slightly higher accuracy at 67%, WOT maintained a similar accuracy of 65%, and GAN-based OT achieved 62%.

Deep Lineage accurately predicts gene expression and early cell fate bias in cellular reprogramming

To evaluate the performance of our model on another dataset with more time points, we applied Deep Lineage to the reprogramming process of fibroblasts into induced endodermal progenitor (iEP) cells over 28 days.²⁰ This dataset comprises gene expression profiles for 18,803 cells and 28,001 genes across six time points (days 6, 9, 12, 15, 21, and 28), with corresponding barcode-derived lineage data at three time points (days 0, 3, and 13) (Figure 5A). Following the same approach as above, we first performed an extensive hyperparameter search to select the parameters that yielded the highest correlation between the held-out samples during training and those predicted by the model (see STAR Methods; Table S3). We then assessed the ability of our model to predict gene expression on different days during the fibroblast transition to iEP cells. Our method consistently has high performance in generating realistic gene expression profiles, regardless of the specific time point chosen for analysis (Figures 5B and S2B). We also compared the gene expression distribution between the model predictions and held-out cells at day 28 and showed that Deep Lineage can accurately predict the full gene expression distribution for both failed and reprogrammed cells (Figures 5C and S4). We also observed strong agreement ($R^2 = 0.98$) between the average predicted gene expression and the actual expression levels for genes in both successfully reprogrammed and failed cells (Figures 5D and 5E).

We evaluated Deep Lineage's performance in predicting the outcome of the reprogramming process (successful or failed reprogramming; Figure 6A) with ground-truth labels and compared it to an established method, CellRank.²¹ After hyperparameter search and training (Tables S4, S5, and S6), the model predicted the day 28 reprogramming outcome from the first four time points (days 6, 9, 12, and 15) with high accuracy (94.6%) (Figure 6B). Receiver operating characteristic (ROC) analysis showed that Deep Lineage outperformed CellRank irrespective of the number of days analyzed (Figure 6C). For both methods, performance improved as more early time points were included, but Deep Lineage performed better in all cases (Figure 6C). To test our model under ablation conditions, we predicted the reprogramming outcome by sampling cells from days 6, 9, 12, 15, and 21 without considering clonal information. As seen for

(C) Correlation of gene expression between Deep Lineage predictions and cells excluded during training, categorized by cell types and time points.

(D and E) Comparison of average gene expression values of 2,000 genes between Deep Lineage predicted and real cells for monocytes and neutrophils. Gene expression at day 4 was averaged across predicted cells per clone and then correlation was computed with actual data for each gene (R^2 indicates the squared Pearson correlation coefficient between the predicted and ground-truth values).

(F and G) Violin plots show gene expression distributions between predicted and real cells for randomly selected genes for both monocytes and neutrophils. Dataset: mouse hematopoiesis (Weinreb et al.¹⁸; GEO: GSE140802), $n = 130,887$ cells and 25,289 genes sampled at days 2, 4, and 6, focusing on the neutrophil/monocyte lineage; clone-level 80%/10%/10% train/validation/test split. In (D) and (E), R^2 is the squared Pearson correlation between predicted and ground-truth mean expression across 2,000 genes. UMAP, uniform manifold approximation and projection. In (C), "Random" is a control in which predicted profiles are compared with randomly sampled cells instead of the matched held-out cells. See also Figure S2 and Table S10.

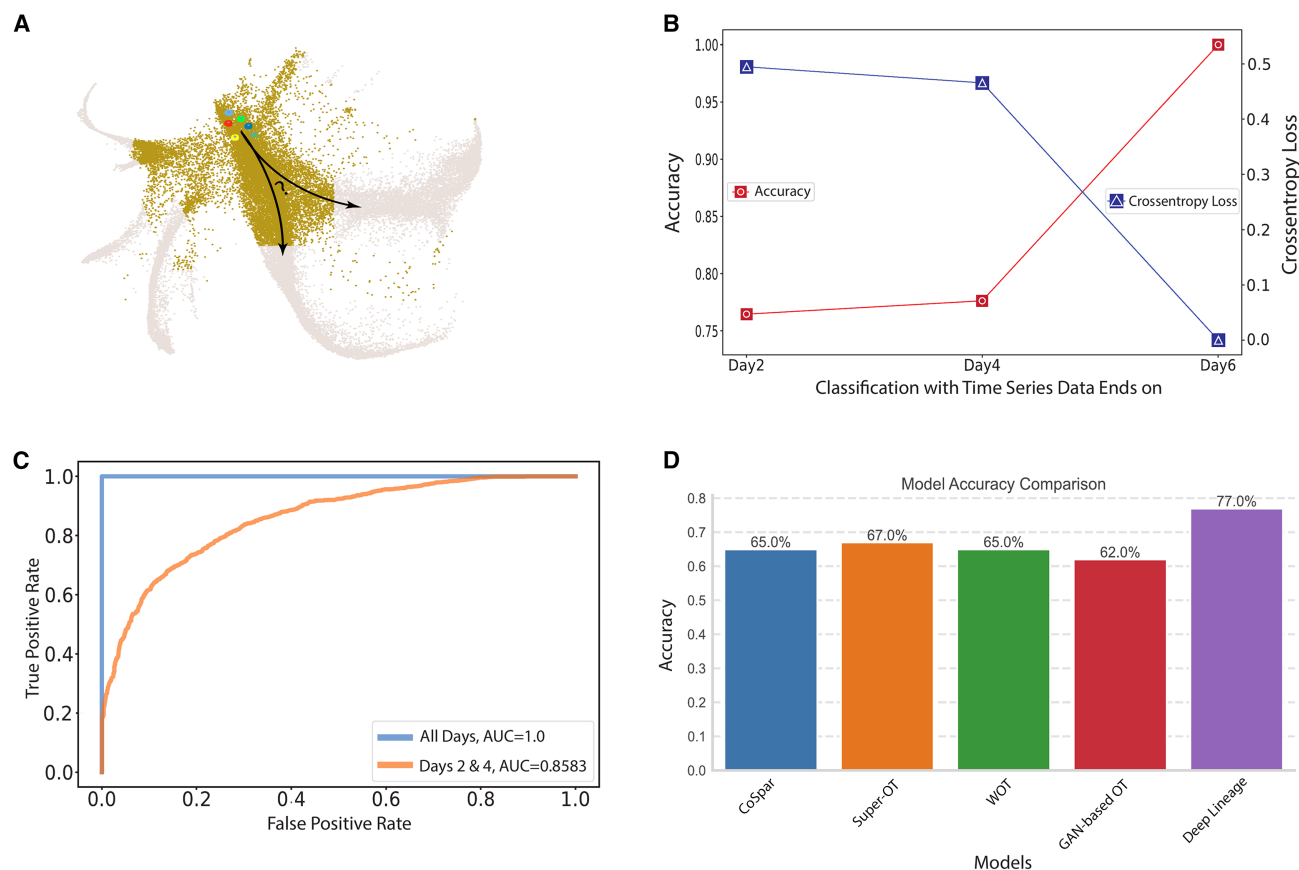


Figure 4. Exploring progenitor bias prediction in hematopoiesis with Deep Lineage and comparison with existing methods

(A) Schematic UMAP diagram illustrating the trajectory of stem cell differentiation and the possible cellular outcomes with cell colors indicating distinct clonal lineages. Gray points represent other mature cell types (not neutrophils and monocytes).

(B) Performance evaluation of the classifier using accuracy and cross-entropy loss metrics to predict early cell fate bias by employing gene expression of cells on just day 2, using cells on days 2 and 4, and including all 3 days within a clone.

(C) Receiver operating characteristic (ROC) curves and area under the curve (AUC) values for the model using cells from days 2 and 4 or all 3 days with a clone, showing our model's performance in early cell fate bias prediction. Day 2 performance is very close to days 2 and 4 performance, thus not plotted.

(D) Comparative analysis with existing methods in predicting early cell fate bias. Deep Lineage outperforms existing methods showing a higher classification accuracy.

Dataset: mouse hematopoiesis (Weinreb et al.¹⁸; GEO: GSE140802). Fate prediction uses a clone-level 80%/10%/10% split with accuracy reported on the 10% held-out test clones; ROC and AUC summarize neutrophil versus monocyte discrimination. Cross-entropy loss is the categorical cross-entropy between the predicted fate-class probabilities and the true labels, for which lower values indicate better classification. In (D), early-fate-prediction accuracies are Deep Lineage 76.8% (shown rounded to 77.0% in the panel), CoSpar 65%, Super-OT 67%, WOT 65%, and GAN-based OT 62%. All methods are evaluated on the same held-out test clones.

See also Figure S3 and Table S2.

the mouse hematopoiesis ablation study, the results showed a decrease in performance, with the best ablated configuration reaching 50.6% and the mean across configurations reaching 50.0%. As in the hematopoiesis analysis, changes to the model architecture (LSTM versus GRU, hyperparameter changes) had a limited impact on accuracy, which remained in the range of approximately 49%–51%. This finding further highlighted the effect of considering clonal structure in accurately predicting reprogramming outcomes (Figure S3B; Table S7).

Finally, to assess the effect of barcoding resolution on model performance within the same biological system, we generated a single barcoding variant of the dataset by constructing lineages using only day 0 barcodes rather than all three barcodes (days 0, 3, and 13) in cumulative barcoding (Figure 6D). On the

same test set, the model trained on single barcoding data achieves 72.9% accuracy, substantially lower than the cumulative barcoding (91.2%). This result confirms that single barcoding lineages suffer from lower lineage resolution, leading to the incorrect grouping of distinct sub-lineages that share identical initial barcodes, highlighting the substantial benefit of higher-resolution barcoding for cell fate prediction.

Deep Lineage identifies candidate driver genes and proposes *in silico* perturbation hypotheses in cellular reprogramming

Detecting and targeting key genes involved in cell reprogramming is critical for advancing our understanding of the system and potential therapeutic applications.²² Deep Lineage uses its

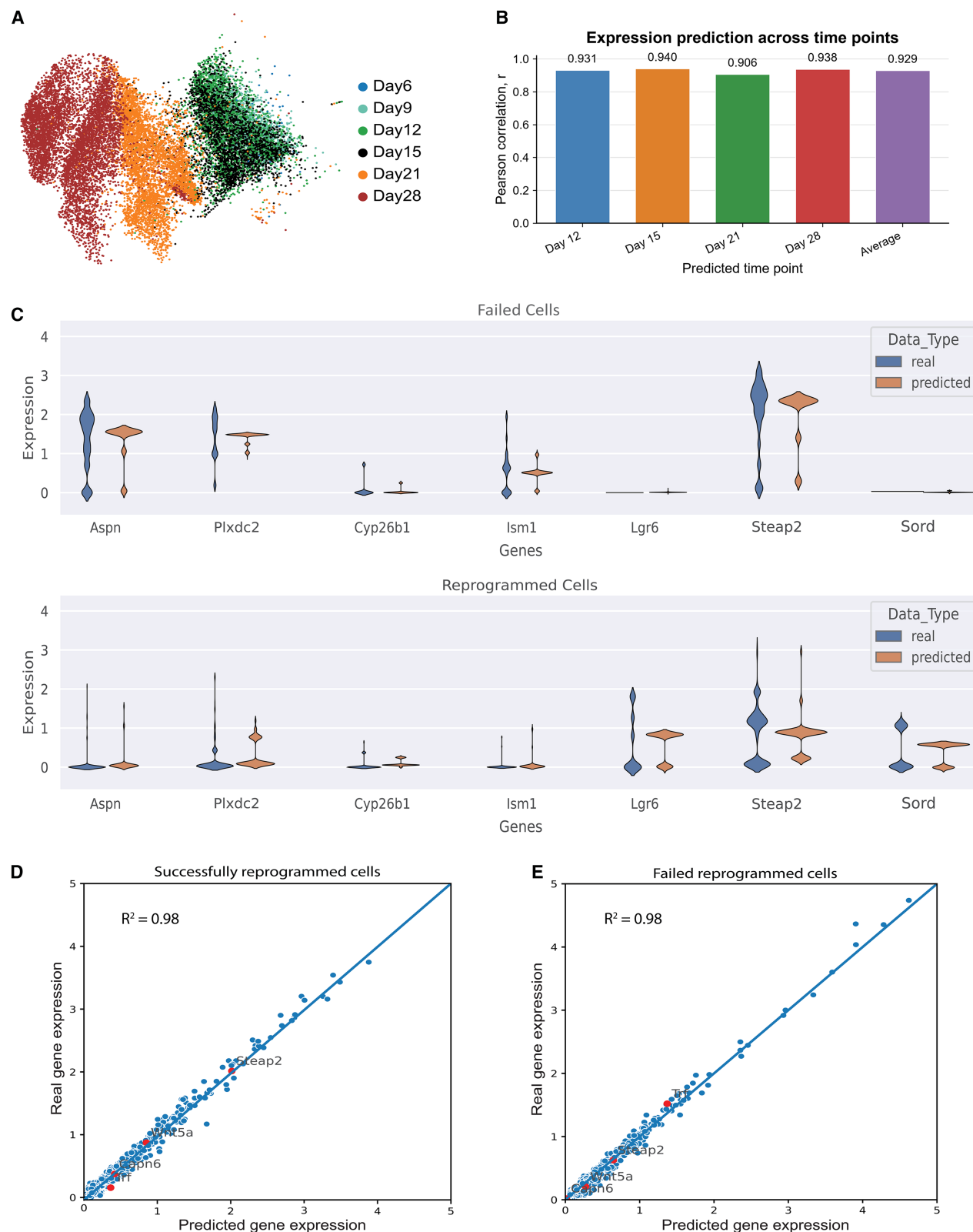


Figure 5. Deep Lineage accurately predicts single-cell gene expression of unseen cells in reprogramming

(A) UMAP visualization demonstrates fibroblast cell reprogramming into iEPs; dots represent cells and are color-coded by time point.

(B) Correlation values between gene expression of ground-truth cells and Deep Lineage predictions for cells excluded during training across different stages of the reprogramming process.

(legend continued on next page)

learned latent representation combined with SHAP analysis to prioritize candidate driver genes within the biological processes. We tested this ability of Deep Lineage by using both fibroblast reprogramming and mouse hematopoiesis data^{18,20} and performed a series of *in silico* perturbations to manipulate the fate of these systems. Using SHAP, we predicted key genes for the reprogrammed and failed trajectories across various time points. Our model independently prioritized established essential marker genes of reprogramming, including the fibroblast identity guardian *Col1a1*, the transition marker *Cryab*, and drivers *Apoa1* and *Sox11*, within its top candidates.^{20,23–25} Furthermore, the model identified *Cdh1* (E-cadherin), a hallmark of the requisite mesenchymal-to-epithelial transition (MET),²⁶ and *Il6st*, a key signaling component for differentiation cytokines,²⁷ highlighting its sensitivity to both structural and signaling effectors. Table S8 presents a curated list of these genes, including their established functions in cellular reprogramming and related processes. The unsupervised rediscovery of these known drivers confirms Deep Lineage's ability to capture biologically relevant regulatory signals without prior knowledge. We also performed gene set enrichment analysis (GSEA) on the top 200 gene candidates demonstrating Deep Lineage's capacity to detect reprogramming-relevant biological pathways such as system and tissue development, proliferation signals, and cellular differentiation (Figure 7C). Together these results suggest that Deep Lineage can be used to identify candidate drivers of reprogramming and differentiation.

An important research goal is to alter cellular fate and reprogram cells *in silico*, redirecting them away from trajectories heading toward undesirable “dead-end” outcomes (Figure 7A). To explore this application, we focused on the top 10 genes identified by the SHAP method in the failed trajectory path (Figure 7D). We created a single perturbation condition with all 10 genes perturbed at once on day 15, overexpressed or knocked out according to their SHAP values. We then applied Deep Lineage to generate gene expression profiles for days 21 and 28 (Figure 7A). Subsequently, we input the generated perturbed time series back into Deep Lineage to evaluate the perturbations' effectiveness in redirecting the fate of the initially failed clones. After the perturbation, the classifier predicts that the failed clones now switch to a successfully reprogrammed state, showing that the model can be used to propose interventions that could drive cellular reprogramming. To further evaluate these predictions, we conducted a correlation analysis comparing the average gene expression of *in silico* reprogrammed cells on day 28 to both real reprogrammed cells and real failed cells. Our results show a substantially higher similarity between the generated *in silico* successful reprogrammed cells

and real successful reprogrammed cells (Pearson correlation = 0.803) compared to real failed cells (0.399) (Figure 7B). This analysis indicates that Deep Lineage generates perturbed cell states that are distinct from the starting state and transcriptomically resemble authentic reprogrammed cells.

We extended this analysis to the hematopoiesis data to identify important genes and simulate the fate switch from neutrophils to monocytes (Figure 7E). This analysis identified key lineage-defining factors, including known drivers such as *Dlk1*, *Gata2*, *Mmp8*, *Elane*, *Lpl*,^{28–32} and the master transcriptional repressor *Gfi1*,³³ which is important for controlling the switch from neutrophils to monocytes (Figure 7F; Table S8). Additionally, the model prioritized *Prtn3* (proteinase 3) and the calprotectin subunit *S100a9*, both definitive markers of neutrophil granule formation and inflammatory function.^{34,35} Using the trained model and performing GSEA on the top 200 identified genes reveals enrichment of pathways associated with immune system development, differentiation, and hematopoiesis (Figure 7G), indicating the model's ability to capture biological pathways relevant to myeloid differentiation. By performing *in silico* knockouts of the top neutrophil-driving genes in progenitor cells on day 2 of clones predestined to become neutrophils, we were successfully able to alter the predicted final fate of the clone, pushing it toward monocytes by day 6. Mirroring *in vivo* phenotypes where the loss of such factors leads to monocyte accumulation, these results demonstrate that Deep Lineage can propose plausible regulatory logic and prioritize perturbation candidates whose predicted effects on gene expression and fate can be tested in follow-up experiments.

DISCUSSION

The analysis of cellular trajectories during differentiation is useful for understanding developmental processes and supporting the development of stem cell therapies. By investigating the routes cells take during these processes, we gain insights into cellular decision-making mechanisms and the timing of critical events.^{36,37} Computational methods for mapping cellular trajectories, such as pseudotime inference, optimal transport, and latent embedding models, only consider gene expression information, even if clonal information is available.^{5,38–42} To consider both gene expression and clonal information in trajectory analysis, we developed Deep Lineage, a method combining autoencoder-based representation learning and LSTM or GRU models for sequential analysis that capture how cells are related to each other within a clonal lineage. Deep Lineage predicts gene expression and early cell fate bias on held-out clones, identifies candidate key genes for cellular reprogramming, and generates

(C) Violin plots show gene expression distributions between real cells and Deep Lineage predictions for randomly selected genes for both failed and reprogrammed cells (top, Failed; bottom, Reprogrammed).

(D and E) Comparative analysis of the top 2,000 differentially expressed genes between predicted and real cells. High correlation values are observed for both successful and failed reprogramming cells, highlighting the accuracy of the model for different cell types (R^2 indicates the squared Pearson correlation coefficient between the predicted and ground-truth values).

Dataset: fibroblast-to-iEP reprogramming (Bidy et al.²⁰; GEO: GSE99915), $n = 18,803$ cells and 28,001 genes across six expression time points (days 6, 9, 12, 15, 21, and 28) with lineage barcodes at days 0, 3, and 13; clone-level 80%/10%/10% split. In (D) and (E), R^2 is the squared Pearson correlation between predicted and ground-truth expression across the top 2,000 differentially expressed genes. iEP, induced endoderm progenitor; UMAP, uniform manifold approximation and projection. In (B), correlations are shown for days 12, 15, 21, and 28, the time points that can be predicted from earlier observations, and “Average” is the mean of these four values.

See also Figures S2 and S4 and Table S3.

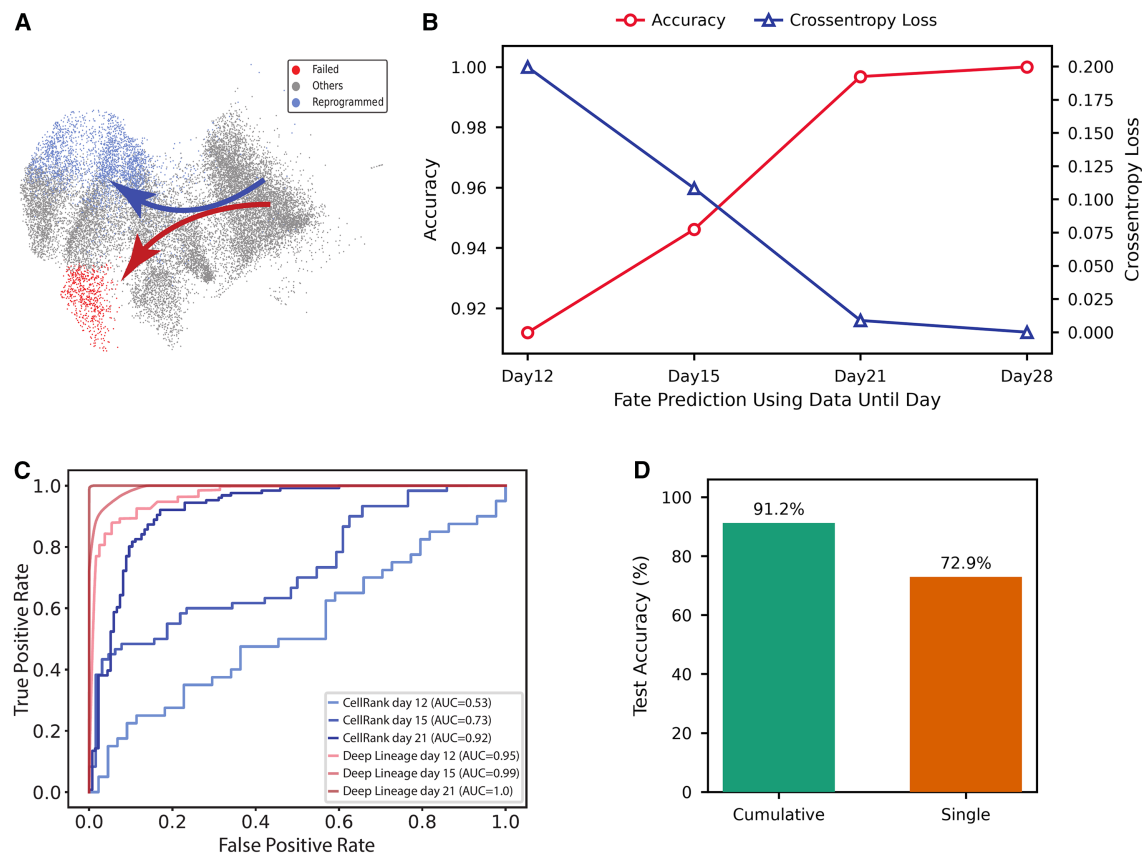


Figure 6. Early cell fate prediction in reprogramming with Deep Lineage and comparison to an existing method

(A) UMAP visualization illustrates progenitor bias toward either successful or failed reprogramming outcomes.

(B) The model's accuracy in detecting successful or failed outcomes of progenitors is examined using gene expression data of cells at all time points up to days 12, 15, 21, and all days within a clone.

(C) Benchmarking comparison between Deep Lineage and CellRank in predicting fate outcome when using all time points up to a given time point (e.g., day 12) to infer time point day 28. Using ROC curves and AUC values, the graph shows the predictive power of both models in predicting fate bias. Deep Lineage consistently outperforms CellRank.

(D) Effect of barcoding resolution on cell fate prediction accuracy. Models are trained on lineages constructed using either cumulative barcoding (days 0, 3, and 13 barcodes) or single barcoding (day 0 barcode only) and evaluated on the same held-out test set. The cumulative barcoding model achieves substantially higher accuracy (91.24%) compared to single barcoding (72.89%), reflecting the increased noise in constructing lineages from lower-resolution barcoding.

Dataset: fibroblast-to-iEP reprogramming (Biddy et al.²⁰; GEO: GSE99915). Accuracy is reported on held-out test clones (clone-level 80%/10%/10% split); ROC and AUC summarize successful versus failed reprogramming discrimination. Cross-entropy loss is the categorical cross-entropy between predicted class probabilities and true labels (lower is better). In (D), cumulative (days 0/3/13) versus single (day 0) barcoding is evaluated on the same held-out test set (91.2% versus 72.9% accuracy).

See also Figure S3 and Tables S4, S5, S6, and S7.

testable *in silico* perturbation hypotheses (e.g., knockout and overexpression) that recover known fate regulators and produce expression states resembling the target lineage. These perturbation outputs are model predictions intended to guide hypothesis generation and prioritize candidates for experimental follow-up.

Deep Lineage implements three neural network architectures to model sequential clonal information: Bi-LSTM, LSTM, and GRU. In our results, among these three architectures, the Bi-LSTM exhibited the highest accuracy for both gene expression prediction and identifying early fate bias but it also had the highest computational cost. LSTM offered accuracy close to Bi-LSTM at lower computational cost, and GRU had the lowest computational cost but sacrificed a small amount of accuracy compared to LSTM. Supporting these three architectures en-

ables researchers to balance accuracy and computational efficiency, facilitating wider adoption in scenarios with limited computational resources. The particular recurrent neural network (RNN) can be selected automatically using hyperparameter optimization or chosen manually based on available computational resources.

Deep Lineage identifies key genes that may be useful to help reprogram cells. However, we cannot prove that our model is causal, as there are not enough time-resolved perturbation experiments available to train or test. Thus, model predictions of key genes are hypothesis generating and can be used with additional information, such as expert knowledge, to select genes likely to be causally implicated in reprogramming processes.

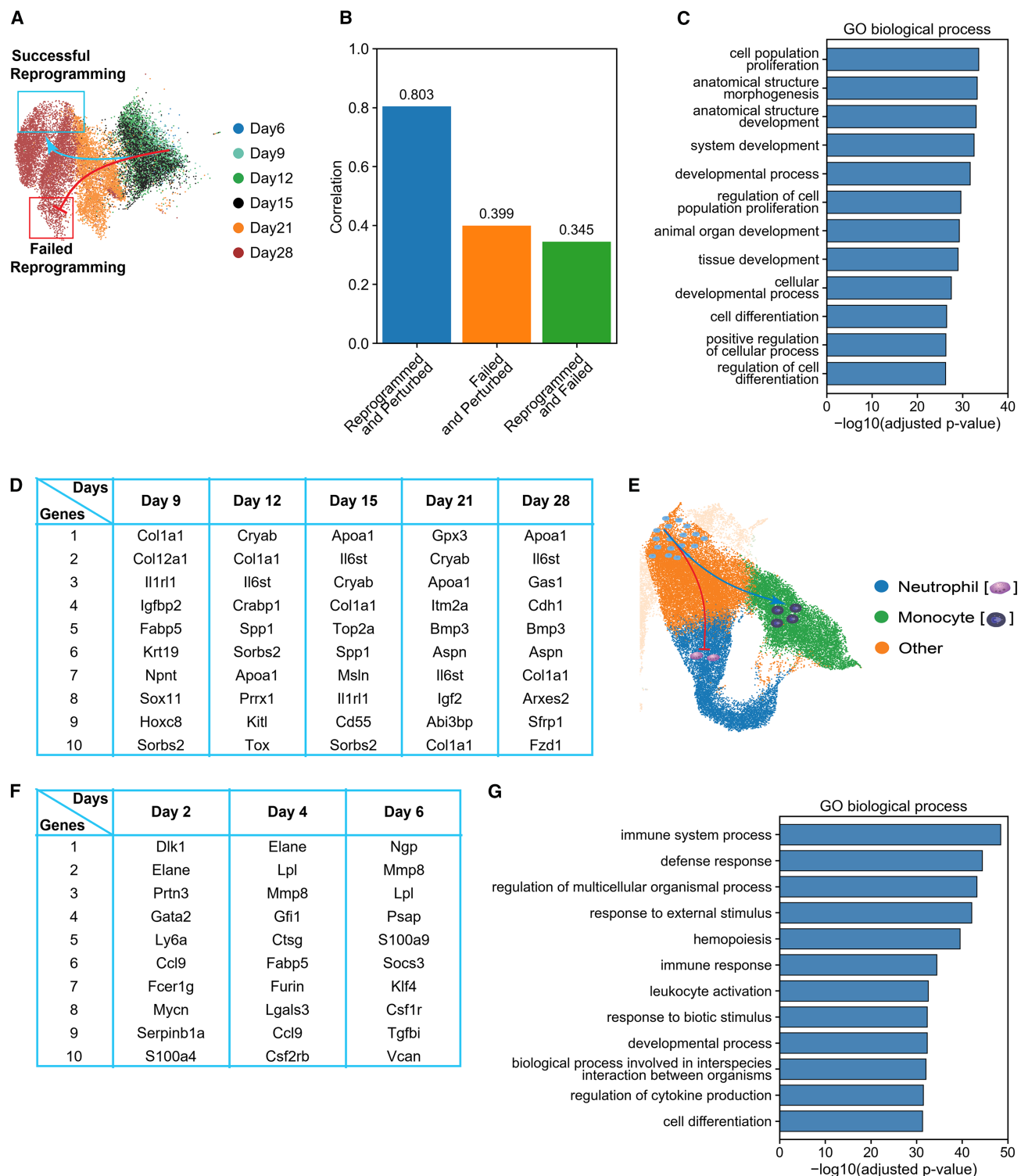


Figure 7. *In silico* perturbations of early cells in hematopoiesis and reprogramming data shift the predicted cell fate at later stages

(A) UMAP representation showing the procedure for *in silico* perturbation of cells on day 15 within failed reprogrammed clones, aiming to switch them into successfully reprogrammed states.

(B) Correlation values between *in silico* perturbed data and actual successfully and unsuccessfully reprogrammed cells on day 28, showing *in silico* perturbed cells are transcriptomically closer to successfully reprogrammed than to failed cells. The right bar (green) compares real reprogrammed with real failed cells as a reference for how similar the two real outcomes are.

(legend continued on next page)

A central challenge in perturbation biology is predicting how a cell responds to a given intervention and understanding how that response unfolds over developmental time. Supervised perturbation-prediction models such as graph-enhanced gene activation and repression simulator (GEARS)⁴³ and scGen⁴⁴ learn a static mapping from a given genetic intervention to a post-perturbation expression profile. These models have no representation of time, cannot distinguish early from late effects of a perturbation, and are unable to model how an initial molecular change cascades through gene regulatory networks, alters fate decisions at successive branchpoints, and reshapes lineage output over days or weeks. In developmental and regenerative biology, this temporal axis is very important, as the same perturbation can produce different outcomes depending on when it is applied and how long the system is allowed to evolve.^{45,46} Deep Lineage addresses this gap by explicitly encoding developmental time into its architecture, which enables it to trace the downstream effects of a perturbation as they accumulate and diverge across differentiation stages.

Current single-cell lineage tracing experiments implement one-step or cumulative barcoding. In one-step barcoding, a single DNA barcode is introduced early and traced over time in a developing system, whereas in cumulative barcoding, new barcodes are generated as the system develops.¹³ We found the inclusion of additional barcoding information in the reprogramming data (cumulative barcoding) can contribute to higher prediction accuracies compared to one-step barcoding. This captures a more comprehensive picture of cellular lineage, leading to enhanced model performance in single-cell trajectory analysis. LSTM and GRU models, with their memory capabilities, are adept at capturing and using sequential dependencies over time, making them well suited to handle the rich, temporal data provided by cumulative barcoding. Consequently, Deep Lineage can achieve higher performance than previous methods in predicting cellular trajectories and early cell fate, benefiting from the detailed lineage information provided by cumulative barcoding. Similar to other computational methods, Deep Lineage is expected to achieve greater accuracy with high-quality, less sparse lineage tracing data, resulting in improved cell fate predictions and gene expression construction. Also, Deep Lineage can open up avenues for future methods. This includes the development of transformer-based models that can be applied to larger lineage tracing and scRNA-seq datasets, potentially improving the scalability and applicability of lineage tracing studies.

We applied Deep Lineage to explore the dynamics of regenerative and developmental processes. However, it should be possible to use it to study other types of cellular plasticity, such as tumor development. Lineage tracing in tumors can be accomplished by inferring clonal relationships between cells based on somatic mutation profiles, as new mutations are generated with every cell division, and this process is often amplified in cancer.⁴⁷ Additionally, Deep Lineage's *in silico* perturbation simulation capability can be used to explore additional treatment strategies and drug interventions. Future work can also expand Deep Lineage by integrating additional data modalities such as epigenomic data. We envision Deep Lineage as a valuable tool for investigating the complex trajectories in regeneration and various developmental processes where the system experiences diverse dynamics.

Several limitations of our approach warrant consideration, particularly for the *in silico* perturbation framework. First, Deep Lineage is trained on unperturbed developmental dynamics, so its perturbation predictions are extrapolations to transcriptional states not observed during training, and their reliability is expected to decrease as the perturbed state significantly departs from the training data distribution. Second, perturbations are applied heuristically, knockedout genes are set to zero and overexpressed genes to five times their baseline, with the top ten candidates perturbed simultaneously, rather than calibrated to physiologically realistic levels, and single-gene perturbations were generally insufficient to completely shift fate. Third, SHAP attributions identify genes influential for the model's predictions, which reflects statistical association within the learned representation rather than proven causal regulation; without large-scale, time-resolved ground-truth perturbation data we cannot confidently establish causality, so the prioritized genes should be regarded as experimentally testable hypotheses. Fourth, the method requires at least three time points with matched single-cell transcriptomes and clonal barcodes, which restricts its applicability to datasets with this design, and its accuracy depends on lineage-readout quality, as we showed performance degrades under barcode dropout and clone misidentification (Figures 2D–2F) and with increasingly sparse data. Finally, part of our benchmarking relies on simulated data (LineageSim), whose generative assumptions may not capture every aspect of real developmental systems, and we treat the cell type and fate labels from the original studies as ground truth. Addressing these limitations will require paired perturbation-lineage datasets and prospective experimental validation of the predicted fate regulators.

(C) Gene set enrichment analysis (GSEA) reveals the biological processes associated with the top 200 genes identified by the SHAP method (40 genes selected from each time point).

(D) Identification of the top 10 gene candidates using SHAP analysis at each time point during fibroblast cell reprogramming.

(E) UMAP visualization depicting the *in silico* perturbation experiment targeting progenitor cells (dark orange) in clones with a neutrophil fate (blue), resulting in a transformation to monocytes (green).

(F) The top 10 gene candidates for hematopoiesis detected by SHAP analysis across all time points.

(G) GSEA of the top 200 genes identified by SHAP analysis.

Datasets: fibroblast-to-iEP reprogramming (Bidy et al.²⁰; GEO: GSE99915; A–D) and mouse hematopoiesis (Weinreb et al.¹⁸; GEO: GSE140802; E–G). Candidate genes are prioritized by SHAP (Shapley additive explanations) values from the trained model; pathway enrichment uses GSEA on the top 200 genes. *In silico* perturbations set knockedout genes to 0 and overexpressed genes to five times baseline, applied to the top 10 genes simultaneously (day 15 for reprogramming and day 2 for hematopoiesis). In (B), similarity is the Pearson correlation coefficient. In (E), dark orange points are perturbed progenitor cells, blue points are neutrophil-fated clones and green points the resulting monocyte fate. MET, mesenchymal-to-epithelial transition. In (C and G), bar length is the $-\log_{10}$ adjusted enrichment p value for one gene set.

See also Table S8.

RESOURCE AVAILABILITY

Lead contact

Further information and requests for resources and reagents should be directed to and will be fulfilled by the lead contact, Mehrshad Sadria (msadria@uwaterloo.ca).

Materials availability

This study did not generate new, unique reagents or materials.

Data and code availability

- scRNA-seq with matched lineage tracing data analyzed in this study are existing, publicly available datasets; their accession numbers (GEO: GSE140802 and GSE99915) are listed in the [key resources table](#).
- All original code has been deposited at GitHub and archived at Zenodo and is publicly available as of the date of publication; GitHub: <https://github.com/garyhlai/deep-lineage>; Zenodo: <https://doi.org/10.5281/zenodo.23570294>.
- Any additional information required to reanalyze the data reported in this paper is available from the [lead contact](#) upon request.

ACKNOWLEDGMENTS

We thank Prof. Anita Layton (University of Waterloo) for helpful discussions and support.

AUTHOR CONTRIBUTIONS

Conceptualization, M.S. and G.D.B.; software, M.S., A.Z., and H.G.L.; methodology, M.S.; validation, M.S., A.Z., and H.G.L.; formal analysis, M.S., A.Z., and H.G.L.; investigation, M.S., A.Z., and H.G.L.; data curation, M.S., A.Z., and H.G.L.; writing, M.S., A.Z., H.G.L., and G.D.B.; visualization, M.S., A.Z., and H.G.L.; supervision, M.S. and G.D.B.; and funding acquisition, G.D.B.

DECLARATION OF INTERESTS

The authors declare no competing interests.

STAR★METHODS

Detailed methods are provided in the online version of this paper and include the following:

- [KEY RESOURCES TABLE](#)
- [METHOD DETAILS](#)
 - Autoencoder for latent embedding of scRNA-seq data
 - RNN models for gene expression and early cell fate prediction
 - Probabilistic formulation and architectural choice (Bidirectional vs. Autoregressive)
 - Autoregressive formulation for cell fate prediction
 - Fate-aware lineage simulation with LineageSim
 - Data preprocessing
 - Modelling choices and training
 - Comparison with existing methods
 - Simulating and evaluating *in silico* perturbations
 - Hardware used
- [QUANTIFICATION AND STATISTICAL ANALYSIS](#)

SUPPLEMENTAL INFORMATION

Supplemental information can be found online at <https://doi.org/10.1016/j.cels.2026.101759>.

Received: May 22, 2024

Revised: April 16, 2026

Accepted: October 1, 2026

REFERENCES

- Griffiths, J.A., Scialdone, A., and Marioni, J.C. (2018). Using single-cell genomics to understand developmental processes and cell fate decisions. *Mol. Syst. Biol.* 14, e8046. <https://doi.org/10.15252/msb.20178046>.
- Haniffa, M., Taylor, D., Linnarsson, S., Aronow, B.J., Bader, G.D., Barker, R.A., Camara, P.G., Camp, J.G., Chédotal, A., Copp, A., et al. (2021). A roadmap for the Human Developmental Cell Atlas. *Nature* 597, 196–205. <https://doi.org/10.1038/s41586-021-03620-1>.
- Weinreb, C., Wolock, S., Tusi, B.K., Socolovsky, M., and Klein, A.M. (2018). Fundamental limits on dynamic inference from single-cell snapshots. *Proc. Natl. Acad. Sci. USA* 115, E2467–E2476. <https://doi.org/10.1073/pnas.1714723115>.
- Saelens, W., Cannoodt, R., Todorov, H., and Saeys, Y. (2019). A comparison of single-cell trajectory inference methods. *Nat. Biotechnol.* 37, 547–554. <https://doi.org/10.1038/s41587-019-0071-9>.
- Tritschler, S., Büttner, M., Fischer, D.S., Lange, M., Bergen, V., Lickert, H., and Theis, F.J. (2019). Concepts and limitations for learning developmental trajectories from single cell genomics. *Development* 146, dev170506. <https://doi.org/10.1242/dev.170506>.
- Chen, C., Liao, Y., and Peng, G. (2022). Connecting past and present: single-cell lineage tracing. *Protein Cell* 13, 790–807. <https://doi.org/10.1007/s13238-022-00913-7>.
- McKenna, A., Findlay, G.M., Gagnon, J.A., Horwitz, M.S., Schier, A.F., and Shendure, J. (2016). Whole-organism lineage tracing by combinatorial and cumulative genome editing. *Science* 353, aaf7907. <https://doi.org/10.1126/science.aaf7907>.
- Schiebinger, G., Shu, J., Tabaka, M., Cleary, B., Subramanian, V., Solomon, A., Gould, J., Liu, S., Lin, S., Berube, P., et al. (2019). Optimal-transport analysis of single-cell gene expression identifies developmental trajectories in reprogramming. *Cell* 176, 1517. <https://doi.org/10.1016/j.cell.2019.02.026>.
- Wang, S.-W., Herriges, M.J., Hurley, K., Kotton, D.N., and Klein, A.M. (2022). CoSpar identifies early cell fate biases from single-cell transcriptomic and lineage information. *Nat. Biotechnol.* 40, 1066–1074. <https://doi.org/10.1038/s41587-022-01209-1>.
- Prasad, N., Yang, K.D., and Uhler, C. (2020). Optimal transport using GANs for lineage tracing. Preprint at arXiv. <https://doi.org/10.48550/arXiv.2007.12098>.
- Forrow, A., and Schiebinger, G. (2021). LineageOT is a unified framework for lineage tracing and trajectory inference. *Nat. Commun.* 12, 4940. <https://doi.org/10.1038/s41467-021-25133-1>.
- VanHorn, S., and Morris, S.A. (2021). Next-generation lineage tracing and fate mapping to interrogate development. *Dev. Cell* 56, 7–21. <https://doi.org/10.1016/j.devcel.2020.10.021>.
- Wagner, D.E., and Klein, A.M. (2020). Lineage tracing meets single-cell omics: opportunities and challenges. *Nat. Rev. Genet.* 21, 410–427. <https://doi.org/10.1038/s41576-020-0223-2>.
- Lai, H.G., and Sadria, M. (2026). LineageSim: a single-cell lineage simulator with fate-aware gene expression. Preprint at bioRxiv. <https://doi.org/10.64898/2026.02.08.704635>.
- Yu, Y., Si, X., Hu, C., and Zhang, J. (2019). A review of recurrent neural networks: LSTM cells and network architectures. *Neural Comput.* 31, 1235–1270. https://doi.org/10.1162/neco_a_01199.
- Cho, K., van Merriënboer, B., Gulcehre, C., Bahdanau, D., Bougares, F., Schwenk, H., and Bengio, Y. (2014). Learning phrase representations using RNN encoder-decoder for statistical machine translation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (Association for Computational Linguistics)*, pp. 1724–1734. <https://doi.org/10.3115/v1/D14-1179>.
- Lundberg, S.M., and Lee, S.-I. (2017). A unified approach to interpreting model predictions. *NIPS'17: Proceedings of the 31st International Conference on Neural Information Processing Systems*. 4768–4777. <https://dl.acm.org/doi/10.5555/3295222.3295230>.

18. Weinreb, C., Rodriguez-Fraticelli, A., Camargo, F.D., and Klein, A.M. (2020). Lineage tracing on transcriptional landscapes links state to fate during differentiation. *Science* 367, eaaw3381. <https://doi.org/10.1126/science.aaw3381>.
19. Yang, K.D., and Uhler, C. (2019). Scalable unbalanced optimal transport using generative adversarial networks. In 7th International Conference on Learning Representations (ICLR 2019). <https://openreview.net/forum?id=HyexAiA5Fm>.
20. Biddy, B.A., Kong, W., Kamimoto, K., Guo, C., Wayne, S.E., Sun, T., and Morris, S.A. (2018). Single-cell mapping of lineage and identity in direct reprogramming. *Nature* 564, 219–224. <https://doi.org/10.1038/s41586-018-0744-4>.
21. Lange, M., Bergen, V., Klein, M., Setty, M., Reuter, B., Bakhti, M., Lickert, H., Ansari, M., Schneringer, J., Schiller, H.B., et al. (2022). CellRank for directed single-cell fate mapping. *Nat. Methods* 19, 159–170. <https://doi.org/10.1038/s41592-021-01346-6>.
22. Wang, H., Yang, Y., Liu, J., and Qian, L. (2021). Direct cell reprogramming: approaches, mechanisms and progress. *Nat. Rev. Mol. Cell Biol.* 22, 410–424. <https://doi.org/10.1038/s41580-021-00335-z>.
23. Luo, C., Lee, Q.Y., Wapinski, O., Castanon, R., Nery, J.R., Mall, M., Kareta, M.S., Cullen, S.M., Goodell, M.A., Chang, H.Y., et al. (2019). Global DNA methylation remodeling during direct reprogramming of fibroblasts to neurons. *eLife* 8, e40197. <https://doi.org/10.7554/eLife.40197>.
24. Chen, Y., Yang, S., Lovisa, S., Ambrose, C.G., McAndrews, K.M., Sugimoto, H., and Kalluri, R. (2021). Type-I collagen produced by distinct fibroblast lineages reveals specific function during embryogenesis and osteogenesis imperfecta. *Nat. Commun.* 12, 7199. <https://doi.org/10.1038/s41467-021-27563-3>.
25. Miao, Q., Hill, M.C., Chen, F., Mo, Q., Ku, A.T., Ramos, C., Sock, E., Lefebvre, V., and Nguyen, H. (2019). SOX11 and SOX4 drive the reactivation of an embryonic gene program during murine wound repair. *Nat. Commun.* 10, 4042. <https://doi.org/10.1038/s41467-019-11880-9>.
26. Li, R., Liang, J., Ni, S., Zhou, T., Qing, X., Li, H., He, W., Chen, J., Li, F., Zhuang, Q., et al. (2010). A mesenchymal-to-epithelial transition initiates and is required for the nuclear reprogramming of mouse fibroblasts. *Cell Stem Cell* 7, 51–63. <https://doi.org/10.1016/j.stem.2010.04.014>.
27. Brady, J.J., Li, M., Suthram, S., Jiang, H., Wong, W.H., and Blau, H.M. (2013). Early role for IL-6 signalling during generation of induced pluripotent stem cells revealed by heterokaryon RNA-Seq. *Nat. Cell Biol.* 15, 1244–1252. <https://doi.org/10.1038/ncb2835>.
28. Li, L., Forman, S.J., and Bhatia, R. (2005). Expression of DLK1 in hematopoietic cells results in inhibition of differentiation and proliferation. *Oncogene* 24, 4472–4476. <https://doi.org/10.1038/sj.onc.1208637>.
29. Castaño, J., Aranda, S., Bueno, C., Calero-Nieto, F.J., Mejia-Ramirez, E., Mosquera, J.L., Blanco, E., Wang, X., Prieto, C., Zabaleta, L., et al. (2019). GATA2 promotes hematopoietic development and represses cardiac differentiation of human mesoderm. *Stem Cell Rep.* 13, 515–529. <https://doi.org/10.1016/j.stemcr.2019.07.009>.
30. Saw, S., Weiss, A., Khokha, R., and Waterhouse, P.D. (2019). Metalloproteases: on the watch in the hematopoietic niche. *Trends Immunol.* 40, 1053–1070. <https://doi.org/10.1016/j.it.2019.09.006>.
31. Liu, C., Han, T., Stachura, D.L., Wang, H., Vaisman, B.L., Kim, J., Klemke, R.L., Remaley, A.T., Rana, T.M., Traver, D., et al. (2018). Lipoprotein lipase regulates hematopoietic stem progenitor cell maintenance through DHA supply. *Nat. Commun.* 9, 1310. <https://doi.org/10.1038/s41467-018-03775-y>.
32. Nasri, M., Ritter, M., Mir, P., Dannemann, B., Aghaallaei, N., Amend, D., Makaryan, V., Xu, Y., Fletcher, B., Bernhard, R., et al. (2020). CRISPR/Cas9-mediated ELANE knockout enables neutrophilic maturation of primary hematopoietic stem and progenitor cells and induced pluripotent stem cells of severe congenital neutropenia patients. *Haematologica* 105, 598–609. <https://doi.org/10.3324/haematol.2019.221804>.
33. Laslo, P., Spooner, C.J., Warmflash, A., Lancki, D.W., Lee, H.-J., Sciammas, R., Gantner, B.N., Dinner, A.R., and Singh, H. (2006). Multilineage transcriptional priming and determination of alternate hematopoietic cell fates. *Cell* 126, 755–766. <https://doi.org/10.1016/j.cell.2006.06.052>.
34. Korkmaz, B., Horwitz, M.S., Jenne, D.E., and Gauthier, F. (2010). Neutrophil elastase, proteinase 3, and cathepsin G as therapeutic targets in human diseases. *Pharmacol. Rev.* 62, 726–759. <https://doi.org/10.1124/pr.110.002733>.
35. Vogl, T., Tenbrock, K., Ludwig, S., Leukert, N., Ehrhardt, C., van Zoelen, M.A.D., Nacken, W., Foell, D., van der Poll, T., Sorg, C., et al. (2007). Mrp8 and Mrp14 are endogenous activators of Toll-like receptor 4, promoting lethal, endotoxin-induced shock. *Nat. Med.* 13, 1042–1049. <https://doi.org/10.1038/nm1638>.
36. Kucinski, I., and Gottgens, B. (2020). Advancing stem cell research through multimodal single-cell analysis. *Cold Spring Harb. Perspect. Biol.* 12, a035725. <https://doi.org/10.1101/cshperspect.a035725>.
37. Sadria, M., and Bury, T.M. (2024). FateNet: an integration of dynamical systems and deep learning for cell fate prediction. *Bioinformatics* 40, btac525. <https://doi.org/10.1093/bioinformatics/btac525>.
38. Ding, J., Sharon, N., and Bar-Joseph, Z. (2022). Temporal modelling using single-cell transcriptomics. *Nat. Rev. Genet.* 23, 355–368. <https://doi.org/10.1038/s41576-021-00444-7>.
39. Bajohr, J., Scott, E.Y., Olfat, A., Sadria, M., Lee, K., Fahim, M., Taha, H.T., Casasbuenas, D.L., Derham, A., Yuzwa, S.A., et al. (2024). Direct lineage conversion of postnatal mouse cortical astrocytes to oligodendrocyte lineage cells. *eLife* 13, RP98632. <https://doi.org/10.7554/eLife.98632>.
40. Sadria, M., and Layton, A. (2025). scVAEDer: integrating deep diffusion models and variational autoencoders for single-cell transcriptomics analysis. *Genome Biol.* 26, 64. <https://doi.org/10.1186/s13059-025-03519-4>.
41. Sadria, M., and Swaroop, V. (2025). Discovering governing equations of biological systems through representation learning and sparse model discovery. *NAR Genom. Bioinform.* 7, lqaf048. <https://doi.org/10.1093/nar-gab/lqaf048>.
42. Sadria, M., Layton, A., Goyal, S., and Bader, G.D. (2024). Fatecode enables cell fate regulator prediction using classification-supervised autoencoder perturbation. *Cell Rep. Methods* 4, 100819. <https://doi.org/10.1016/j.crmeth.2024.100819>.
43. Roohani, Y., Huang, K., and Leskovec, J. (2024). Predicting transcriptional outcomes of novel multigene perturbations with GEARS. *Nat. Biotechnol.* 42, 927–935. <https://doi.org/10.1038/s41587-023-01905-6>.
44. Lotfollahi, M., Wolf, F.A., and Theis, F.J. (2019). scGen predicts single-cell perturbation responses. *Nat. Methods* 16, 715–721. <https://doi.org/10.1038/s41592-019-0494-8>.
45. Schlaeger, T.M., Mikkola, H.K.A., Gekas, C., Helgadottir, H.B., and Orkin, S.H. (2005). Tie2Cre-mediated gene ablation defines the stem-cell leukemia gene (SCL/tal1)-dependent window during hematopoietic stem-cell development. *Blood* 105, 3871–3874. <https://doi.org/10.1182/blood-2004-11-4467>.
46. Teague, S., Primavera, G., Chen, B., Liu, Z.-Y., Yao, L., Freeburne, E., Khan, H., Jo, K., Johnson, C., and Heemskerk, I. (2024). Time-integrated BMP signaling determines fate in a stem cell model for early human development. *Nat. Commun.* 15, 1471. <https://doi.org/10.1038/s41467-024-45719-9>.
47. Yang, D., Jones, M.G., Naranjo, S., Rideout, W.M., Min, K.H.J., Ho, R., Wu, W., Replogle, J.M., Page, J.L., Quinn, J.J., et al. (2022). Lineage tracing reveals the phylogenetics, plasticity, and paths of tumor evolution. *Cell* 185, 1905–1923.e25. <https://doi.org/10.1016/j.cell.2022.04.015>.
48. Abadi, M., Barham, P., Chen, J., Chen, Z., Davis, A., Dean, J., Devin, M., Ghemawat, S., Irving, G., Isard, M., et al. (2016). TensorFlow: a system for large-scale machine learning. In *Proceedings of the 12th USENIX Symposium on Operating Systems Design and Implementation*, pp. 265–283.
49. Wolf, F.A., Angerer, P., and Theis, F.J. (2018). SCANPY: large-scale single-cell gene expression data analysis. *Genome Biol* 19, 15. <https://doi.org/10.1186/s13059-017-1382-0>.

50. Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., et al. (2011). Scikit-learn: machine learning in Python. *J. Mach. Learn. Res.* **12**, 2825–2830.
51. Harris, C.R., Millman, K.J., van der Walt, S.J., Gommers, R., Virtanen, P., Cournapeau, D., Wieser, E., Taylor, J., Berg, S., Smith, N.J., et al. (2020). Array programming with NumPy. *Nature* **585**, 357–362. <https://doi.org/10.1038/s41586-020-2649-2>.
52. Virtanen, P., Gommers, R., Oliphant, T.E., Haberland, M., Reddy, T., Cournapeau, D., Burovski, E., Peterson, P., Weckesser, W., Bright, J., et al. (2020). SciPy 1.0: fundamental algorithms for scientific computing in Python. *Nat. Methods* **17**, 261–272. <https://doi.org/10.1038/s41592-019-0686-2>.
53. Hunter, J.D. (2007). Matplotlib: a 2D graphics environment. *Comput. Sci. Eng.* **9**, 90–95. <https://doi.org/10.1109/MCSE.2007.55>.
54. Waskom, M.L. (2021). seaborn: statistical data visualization. *J. Open Source Softw.* **6**, 3021. <https://doi.org/10.21105/joss.03021>.
55. Zhai, J., Zhang, S., Chen, J., and He, Q. (2018). Autoencoder and its various variants. In 2018 IEEE International Conference on Systems, Man, and Cybernetics, pp. 415–419. <https://doi.org/10.1109/SMC.2018.00080>.
56. Young, T., Hazarika, D., Poria, S., and Cambria, E. (2018). Recent trends in deep learning-based natural language processing [review article]. *IEEE Comput. Intell. Mag.* **13**, 55–75. <https://doi.org/10.1109/mci.2018.2840738>.
57. Pan, X., Li, H., and Zhang, X. (2022). TedSim: temporal dynamics simulation of single-cell RNA sequencing data and cell division history. *Nucleic Acids Res.* **50**, 4272–4288. <https://doi.org/10.1093/nar/gkac235>.
58. Ahlmann-Eltze, C., and Huber, W. (2023). Comparison of transformations for single-cell RNA-seq data. *Nat. Methods* **20**, 665–672. <https://doi.org/10.1038/s41592-023-01814-1>.

STAR★METHODS

KEY RESOURCES TABLE

REAGENT or RESOURCE	SOURCE	IDENTIFIER
Deposited data		
Mouse hematopoiesis scRNA-seq and lineage tracing (LARRY)	Weinreb et al. ¹⁸	GEO: GSE140802
Fibroblast-to-iEP reprogramming scRNA-seq and lineage tracing (CellTagging)	Biddy et al. ²⁰	GEO: GSE99915
Preprocessed hematopoiesis dataset	Wang et al. ⁹ (CoSpar)	https://cospar.readthedocs.io/
Software and algorithms		
Deep Lineage (this study)	This paper	GitHub: https://github.com/garyhlai/deep-lineage ; Zenodo: https://doi.org/10.5281/zenodo.23570294
Python 3.9.13	Python Software Foundation	https://www.python.org/
TensorFlow-GPU 2.10.0; Keras 2.10.0	Abadi et al. ⁴⁸	https://www.tensorflow.org/
scanpy 1.8.2	Wolf et al. ⁴⁹	https://scanpy.readthedocs.io/
scikit-learn 1.1.1	Pedregosa et al. ⁵⁰	https://scikit-learn.org/
NumPy 1.22.4; SciPy 1.10.1; pandas 1.4.3	Harris et al. ⁵¹ and Virtanen et al. ⁵²	https://numpy.org/
matplotlib 3.5.1; seaborn 0.12.2	Hunter ⁵³ and Waskom ⁵⁴	https://matplotlib.org/
CoSpar 0.2.1	Wang et al. ⁹	https://cospar.readthedocs.io/
CellRank	Lange et al. ²¹	https://cellrank.readthedocs.io/
SHAP	Lundberg and Lee ¹⁷	https://github.com/shap/shap

METHOD DETAILS

Autoencoder for latent embedding of scRNA-seq data

To achieve the reduced dimension of the data before training the LSTM model, we employed an autoencoder, an unsupervised neural network algorithm.⁵⁵ The autoencoder consists of two components: an encoder, which compresses the high-dimensional scRNA-seq data into a lower-dimensional latent space, and a decoder, which reconstructs the data back to the original input space. The objective of the autoencoder is to minimize the reconstruction error, measured using the mean squared error (MSE) loss function, expressed as:

$$L_{AE} = \frac{1}{N} * \sum_{i=1}^N \|x_i - \text{Decoder}(\text{Encoder}(x_i))\|$$

Where N is the number of training samples, x_i is the i-th input data sample which is the gene expression of a cell. Encoder() and Decoder() represent the encoder and the decoder parts of the autoencoder. By minimizing the above loss, the autoencoder learns a lower-dimensional representation of the data while preserving important biological features, facilitating the analysis of time series models (LSTMs, GRU).

RNN models for gene expression and early cell fate prediction

For the regression task of predicting gene expression for cells within a clone, we harnessed the power of RNNs, mainly LSTMs and GRU. Unlike traditional feedforward neural networks, LSTM networks have loops and gates that enable information to persist and flow across time steps, making them well-suited for tasks involving time series and sequential data analysis. The distinguishing feature of LSTM is its memory block, which maintains a hidden state and a block state. The block state acts as a long-term memory, enabling the network to keep and update relevant information over long time ranges. The hidden state carries short-term memory and interacts with the block state through various gates. The LSTM block is composed of three main gates that control the flow of information: the input gate (i), the forget gate (f), and the output gate (o).⁵⁶ The input gate decides how much of the new input should be added to the block state. The forget gate controls what information from the previous block state should be discarded, allowing the network to selectively remember or forget information. The output gate determines what part of the block state should be exposed as the hidden state. The equations governing the LSTM block operations are as follows:

Input gate i_t :

$$i_t = \sigma(W_i * h_{t-1} + U_i * X_t + b_i)$$

Forget gate f_t :

$$f_t = \sigma(W_f * h_{t-1} + U_f * X_t + b_f)$$

Output gate o_t :

$$o_t = \sigma(W_o * h_{t-1} + U_o * X_t + b_o)$$

Block state c_t :

$$c_t = f_t * c_{t-1} + i_t * \tanh(W_c * h_{t-1} + U_c * X_t + b_c)$$

Hidden state h_t :

$$h_t = o_t * \tanh(c_t)$$

Here, X_t represents the input at the current time step, h_{t-1} denotes the hidden state from the previous time step, and σ is the sigmoid activation function. The weights and biases are represented by $W_i, W_f, W_o, W_c, U_i, U_f, U_o$, and U_c , are learnable parameters that the LSTM network optimizes during training. To obtain a compact representation of the gene expression data, we used the latent layer of an autoencoder as input to the LSTM model.

GRU is an RNN variant, designed to address some of the limitations of traditional RNNs, such as the vanishing gradient problem that can interfere with the learning process. Similar to LSTM, GRU is particularly effective in processing sequential data and capturing dependencies over long sequences. Each GRU block also contains a hidden state, but instead of having separate blocks and hidden states like LSTM, it uses an update gate (z) and a reset gate (r) to control the flow of information.⁵⁶ These gates enable the GRU to selectively update and reset the hidden state, facilitating the retention of relevant information and reducing the number of parameters compared to LSTM. In GRUs, the reset gate controls how much of the previous hidden state should be forgotten, allowing the network to decide what information to retain from the past. The update gate determines how much of the candidate hidden state h'_t should be blended with the previous hidden state h_{t-1} to generate the new hidden state h_t . This gate mechanism enables the GRU to regulate the information flow effectively. The operations of a GRU block can be summarized as follows:

Reset gate r_t :

$$r_t = \sigma(W_r * h_{t-1} + U_r * X_t + b_r)$$

Update gate z_t :

$$z_t = \sigma(W_z * h_{t-1} + U_z * X_t + b_z)$$

Candidate hidden state h'_t :

$$h'_t = \tanh(W_h * (r_t * h_{t-1}) + U_h * X_t + b_h)$$

Hidden state h_t :

$$h_t = (1 - z_t) * h_{t-1} + z_t * h'_t$$

Here, X_t represents the input at the current time step, h_{t-1} denotes the hidden state from the previous time step, and σ is the sigmoid activation function. The weights and biases are represented by $W_r, W_z, W_h, U_r, U_z, U_h, b_r, b_z$, and b_h are learnable parameters that the GRU network optimizes during training. GRU is computationally more efficient than LSTM due to its simplified architecture, making it particularly suitable for handling extensive datasets with limited computational resources or settings with limited training data.

The equations mentioned earlier can be used for both regression and classification tasks. In classification tasks, the model's output is then connected to a fully connected layer that handles the classification. Employing softmax activation in this layer allows the model to derive class probabilities.

$$\text{Classification_output} = \text{Softmax}(w * h_t + b)$$

Probabilistic formulation and architectural choice (Bidirectional vs. Autoregressive)

We formulate the cell fate prediction problem as a conditional probability estimation over a temporal sequence. While this can be modeled using a standard autoregressive approach (predicting the next state x_t based solely on past states $x_{<t}$), similar to language models, we found that lineage analysis benefits substantially from access to future context during training. Lineage tracing data uniquely provides clone-matched endpoints, enabling us to leverage information from both progenitors and descendants. Therefore, instead of a strictly autoregressive formulation, we employ a bidirectional architecture (Bi-LSTM) trained via a masking objective. This approach enables the model to condition on the full temporal context $P(x_t | x_{<t}, x_{>t})$, enabling it to perform intermediate interpolation tasks that are mathematically inaccessible to unidirectional autoregressive models. A detailed theoretical derivation of the autoregressive formulation and its comparison to our bidirectional approach is provided below ([autoregressive formulation for cell fate prediction](#)).

Autoregressive formulation for cell fate prediction

As an alternative to the bidirectional design, we explored an autoregressive formulation that predicts terminal cell fate by sequentially generating gene expression along a lineage, followed by terminal cell-type prediction. The approach consists of two independently trained components: (i) a cell-type classifier, a feedforward neural network mapping gene expression directly to cell type; and (ii) an autoregressive gene-expression regressor, a multi-layer LSTM trained in a sequence-to-sequence fashion to predict future gene expression. We train the LSTM with teacher forcing on temporally shifted gene-expression sequences: given expression at time points x_0 and x_1 , the model is trained to predict expression at x_1 and x_2 , respectively.

At inference we evaluate three increasingly challenging scenarios: (1) direct classification, in which x_2 expression is provided directly to the classifier; (2) forward prediction, in which x_2 expression is predicted from (x_0, x_1) using the LSTM and then classified; and (3) autoregressive prediction, in which x_1 is first predicted from x_0 , then x_2 is predicted from $(x_0, \text{predicted } x_1)$, followed by classification. The classifier is a four-layer multilayer perceptron with hidden dimensions [512, 256, 128], ReLU activations, dropout (0.3), batch normalization, L2 regularization ($1e-5$), and a sigmoid output trained with binary cross-entropy loss; the autoregressive regressor is a three-layer LSTM with hidden dimension 256 and dropout 0.3.

Using LineageSim data (2,000 genes; 4,106 trajectories; 8,192 cells), the autoregressive model achieves 78.5% accuracy when predicting terminal fate from x_0 alone and 89.1% when x_1 expression is also provided, whereas direct classification from x_2 expression achieves perfect accuracy as an upper bound (Table S11). Although autoregressive models can be effective for cell fate prediction, they are inherently limited to causal, forward-only generation and cannot perform bidirectional inference, such as reconstructing intermediate states from earlier and later observations (for example, predicting x_1 from x_0 and x_2). Deep Lineage is explicitly designed to support both forward prediction and backward imputation within a unified framework, which motivated our choice of the bidirectional architecture.

Fate-aware lineage simulation with LineageSim

To benchmark cell fate inference under known ground truth, we developed LineageSim,¹⁴ a fate-aware lineage simulator for single-cell gene expression data. Unlike existing simulators, which generate gene expression conditioned only on the immediate cell state and therefore carry no long-range coupling between progenitor expression and descendant fate (Markovian), LineageSim injects a long-range fate signal into the latent representations that determine expression, so that each progenitor's transcriptome carries some information about its descendants' terminal fates, the feature that fate-prediction methods are designed to recover. LineageSim first constructs a phylogenetic tree by stochastic binary cell division, which produces a fully resolved tree across discrete generations. Terminal fates (0 or 1) are then assigned probabilistically to cells at a configurable commitment depth D under a user-specified class prior (default 50/50) and propagated to all descendants. For every progenitor at depth $d \leq D$ we compute a fate bias $p \in [0, 1]$ defined as the fraction of its committed descendants assigned to Fate 1, giving a continuous summary of the progenitor's eventual developmental outcome. Gene expression is then driven by a per-cell Cell Identity Factor (CIF) latent vector that evolves recursively along the tree under a fate-dependent drift plus Brownian noise:

$$\text{CIF} = \text{CIF}_{\text{parent}} + \beta \cdot M(d) \cdot (2p - 1) + \varepsilon, \varepsilon \sim N(0, \sigma^2),$$

where β sets the strength of the fate signal, σ the per-step diffusion, and $(2p - 1)$ acts as a directional force (progenitors enriched for Fate 1, $p > 0.5$, drift positively; those enriched for Fate 0 drift negatively). The modulator $M(d)$ reflects that fate commitment strengthens with differentiation, taking the values 0.4 for $d \leq \lfloor 3D/7 \rfloor$ (early progenitors, weak signal), 0.8 for $\lfloor 3D/7 \rfloor < d \leq D$ (intermediate commitment), and 1.2 for $d > D$ (committed cells, strong signal). Each gene is mapped from the CIF to two-state kinetic parameters (burst frequency, burst size, and mRNA degradation rate) through sparse weight vectors, and count-level expression is sampled from the kinetic model used by TedSim,⁵⁷ to which technical noise is added to mimic scRNA-seq dropout. For the benchmarks reported in this study we generated a tree of 8192 cells across two terminal fates with 2000 simulated genes, fate-signal strength $\beta = 1.0$, Brownian noise $\sigma = 0.5$, commitment depth $D = 9$, and a 50/50 class prior, from which we sampled 4106 cell lineage trajectories. Each trajectory consists of three temporally ordered cells: an early progenitor x_0 , an intermediate cell x_1 , and a terminal cell x_2 all drawn from the same root-to-leaf path through the lineage tree. The trajectory-level fate distribution was nearly balanced (2051 vs 2055). A classifier baseline showing that the injected fate signal is recoverable from progenitor expression above chance level is reported in 14. All splits on LineageSim data are performed at the trajectory (clone) level: trajectories are randomly partitioned into 80% training, 10% validation, and 10% test, with all three cells of a trajectory remaining within the same fold. Because each trajectory is a distinct root-to-leaf path through the lineage tree, this prevents leakage of cell-of-origin information between folds. The same trajectory-level split is used for the barcode-dropout and clone-misidentification robustness experiments (Figure 2). Test trajectories are held fixed across all noise conditions for fair comparison, and dropout or label-shuffling is applied only within the training and validation folds. For the real datasets, the analogous split is performed at the clone level so that all cells sharing a clonal barcode reside in a single fold, again preventing leakage of lineage information across different sets.

Data preprocessing

To show Deep Lineage can handle different barcoding strategies, we applied it to data generated from two barcoding methods: Larry and CellTagging.^{18,20} Given the selection of a preprocessing method depends on the specific goal of the analysis, the performance of a supervised method can serve as a useful metric. Therefore, in each task (prediction of cell fate and gene expression), we compare

the performance of different preprocessing steps before training the models and choose the one that leads to higher accuracy (lower reconstruction error for AE and lower loss for RNN) for that data. Our evaluation involves many combinations that incorporate different count normalization, log transformation, several scaling techniques, different numbers of highly variable genes, and different dimensionality reductions. For both reprogramming and hematopoiesis data, we found that log transformation without count normalization achieves the highest accuracy and as expected using only raw counts or just scaling without log transformation results in poor performance which is consistent with a recently published scRNA-seq data processing benchmark paper⁵⁸ (Table S9). Following this, we select the highly variable genes and employ an autoencoder for dimensionality reduction (the architectures can be found in Table S9). We used these two datasets because they are based on two distinct lineage-tracing methodologies, LARRY and CellTagging, which link cell states across time rather than only at a single endpoint, do not require lineage-tree inference to establish sister-cell relationships, exhibit low single-cell barcode-dropout rates, and need only a lentiviral barcode library rather than a multi-component genome-editing system.

Modelling choices and training

After completing the preprocessing step, the gene expression data is partitioned into three subsets: 80% for training, 10% for validation, and 10% for testing. To optimize model performance, we conduct an extensive hyperparameter search using various model types (LSTM, GRU, bidirectional LSTM), model architecture (number of layers, nodes in each layer), and activation function (tanh, ReLU, leaky ReLU with $\alpha = 0.3$) (the detailed results are provided in Tables S9 and S10). Subsequently, we select specific sets of parameters that demonstrate higher accuracy for both the autoencoder reconstruction and the LSTMs during regression and classification tasks, tailored to each dataset. For the autoencoder, we employ the mean squared error as the loss function, to ensure effective learning of the latent representation. After training the autoencoder, the resulting latent embedding is used to train the RNN models. In the regression task, where we aim to predict gene expression levels of cells across different developmental stages, we employ the mean squared error as the loss function. For predicting early cell fate bias (classification task), we adopt the categorical cross-entropy loss function. Throughout the training process of both the autoencoder and the RNN models, we implement early stopping, which enables us to halt the training once the models demonstrate optimal performance on the validation set, preventing overfitting and supporting generalization to unseen data. In each iteration, we randomly select a training sample and update the network's weights using SGD (Stochastic Gradient Descent) and Adam optimization algorithms with a default learning rate of 0.001. Also, RNN training can be slow due to the sequential nature of computations. To tackle this challenge, we use the cuDNN LSTM technique, which uses GPU-accelerated optimization to enhance the training efficiency of Long Short-Term Memory (LSTM) networks. By using the parallel processing capabilities and specialized algorithms offered by cuDNN LSTM, we achieve a substantial acceleration in model training speed. In addressing instances of missing data within time series, a common approach involves the use of masking techniques. When a data point at a specific time step is not available, the model skips that particular time-step and proceeds to the subsequent one. By using masking in both classification and regression tasks, we ensure that the time series model handles missing data consistently and effectively. Regarding the model selection, we observe that GRU and LSTMs achieve high accuracies with LSTMs showing slightly higher performance (Tables S3, S4, S5, and S9). However, if computational resources are limited, we recommend using GRU over LSTMs, as GRU requires less computing power.

Comparison with existing methods

To establish a benchmark for evaluating the performance of Deep Lineage in early cell fate bias and predicting gene expression of cell progenies, for hematopoiesis data we selected existing relevant methods to compare to: Cospar9, Super-OT,¹⁰ WOT,⁸ and GAN-Based OT.¹⁹ We adopted the same task as Super OT,¹⁰ which involves evaluating the accuracy of fate prediction for day 2 cells in relation to their development into day 4 and 6 cell types. Specifically, using the clonal information from the dataset, we identify whether each day 2 cell is more likely to differentiate into a neutrophil or a monocyte, assigning this label accordingly. For Cospar and WOT, we used the default parameters as reported in their documentation. For other methods where the code was not available, we attempted to reimplement the method. In cases where our reimplementation resulted in lower accuracy, we used the accuracy metrics reported in their respective studies to ensure a fair comparison of our model's performance with these established benchmarks.

In the context of early cell fate bias in the reprogramming dataset, we conducted a comparative analysis between Deep Lineage and CellRank.²¹ The CellRank framework integrates similarity-based trajectory inference with RNA velocity to uncover directed probabilistic state-change trajectories. When applying CellRank to the reprogramming data, we followed the recommended preprocessing steps, including filtering genes based on a minimum expression threshold across ten cells and at least 20 counts in both spliced and unspliced layers. Following this, normalization by total counts per cell was performed, along with log transformation, while selecting the top 2,000 highly variable genes. Subsequently, a PCA representation was computed using the top 30 principal components (PCs) to construct a K-nearest neighbour (KNN) graph, with K set to 30.

Simulating and evaluating *in silico* perturbations

Model perturbation experiments were conducted using both regression and classification models. *In silico* perturbations were introduced by modifying the preprocessed expression values of target genes in the input data (cell/clone/time-point-by-gene matrix). For simulated gene knockdown, expression was set to zero, while for overexpression, it was set to five times the baseline expression level. These perturbations were applied at specific time points in the experimental timeline: day 15 for the reprogramming dataset

and day 2 for the hematopoiesis dataset. Single-gene perturbations were not sufficient to see strong effects, so we perturbed all 10 genes simultaneously, as described in the results.

Post-perturbation, we used the altered time-series data as input for our regression models, which were then employed to predict the gene expression data for subsequent time points, specifically day 21 and day 28 for reprogramming and day 4 and day 6 for hematopoiesis. For consistency, the same neural network architecture as the baseline model (trained on the original data before perturbation) was used for each dataset. Following generation of the perturbed time-series data, we used the classifier from our clonal fate-bias experiments to classify clone fates. The focus of these perturbations was on monocytes and neutrophils, motivated by available experimental evidence or reported associations for these cell types. The target genes for perturbation were selected based on the output of the SHAP method, which ranked gene importance within the model and guided selection of genes with a strong impact on the fate of monocytes and neutrophils.

Hardware used

Three computational setups are used for data processing, model training, and plotting. A distributed computing system was used extensively for data processing and model training. Nvidia A100 (40GB) and Nvidia P100 (16GB) GPUs were used on this computing server. Intel Xeon E5-2680v4 2.4 GHz (Broadwell) and AMD EPYC 7542 2.9 GHz CPUs were used for other tasks. The maximum memory used on these servers was 64GB. Additionally, the SciNet Trillium cluster was used for model training. A single Nvidia H100 (80GB) GPU and AMD "Genoa" 2.4 GHz CPUs were used on this cluster, with up to 768 GB of RAM available per node.

QUANTIFICATION AND STATISTICAL ANALYSIS

Model performance was quantified as follows. Gene-expression prediction was evaluated using the Pearson correlation coefficient (r) between predicted and held-out ground-truth expression, computed per gene or per clone as indicated in each figure legend; R^2 denotes the squared Pearson correlation. Cell-fate classification was evaluated using accuracy, categorical cross-entropy loss (the cross-entropy between the predicted class-probability distribution and the true fate labels, for which lower values indicate better classification), and the area under the Receiver Operating Characteristic curve (AUC). For every dataset, clones or, for simulated data, trajectories were partitioned into 80% training, 10% validation and 10% test, with all cells of a clone assigned to the same fold to prevent leakage of lineage information, and accuracies are reported on the held-out test set. Candidate fate-driver genes were prioritized using SHAP (SHapley Additive exPlanations) values from the trained model, and enrichment of the top 200 genes was assessed by Gene Set Enrichment Analysis (GSEA). The numbers of cells, clones, trajectories and genes underlying each analysis are stated in the corresponding figure legends. Throughout, statistical terminology is used only in its formal statistical sense, comparative statements not supported by a formal statistical test are described using non-statistical language. Unless a legend states otherwise, each value plotted in a figure is a single summary statistic computed on the held-out set for one condition rather than a summary of replicate measurements, so inferential tests are not reported and comparisons between conditions are descriptive.

Cell Systems, Volume 17

Supplemental information

**Deep Lineage: Single-cell lineage tracing
and fate inference using deep learning**

Mehrshad Sadria, Allen Zhang, Haizhi Gary Lai, and Gary D. Bader

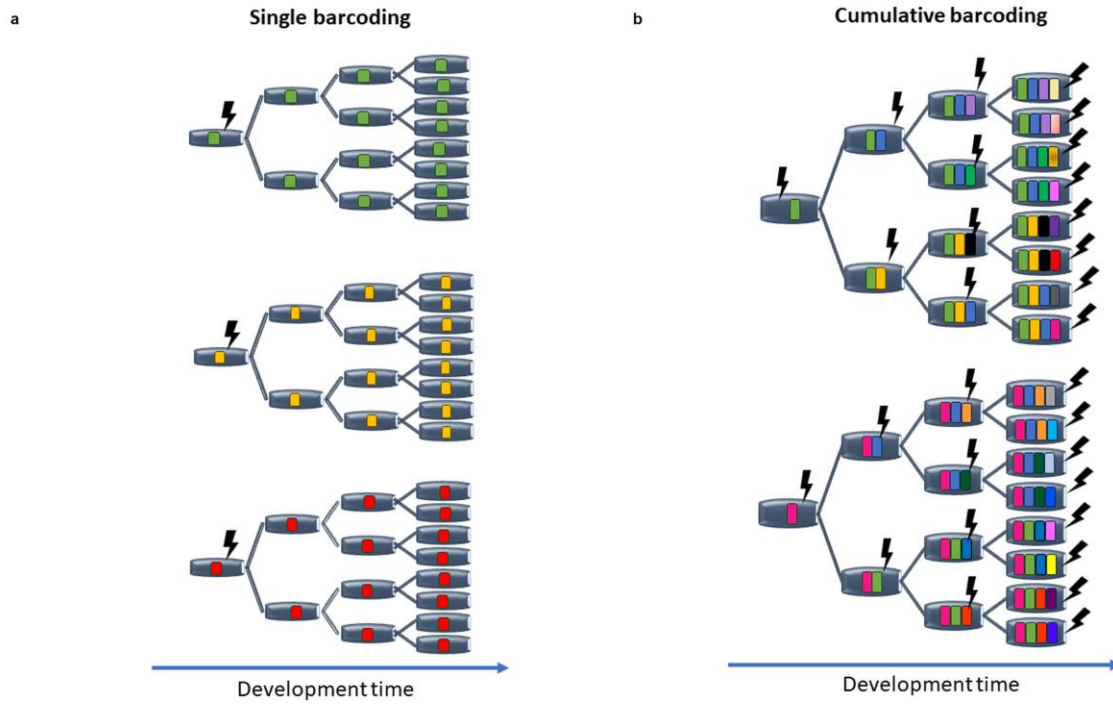


Figure S1. Deep Lineage is compatible with both single and cumulative barcoding data, related to Figure 1. (a) Illustrates single barcoding, while (b) presents a cumulative barcoding strategy. This figure is a schematic illustration and does not display experimental data.

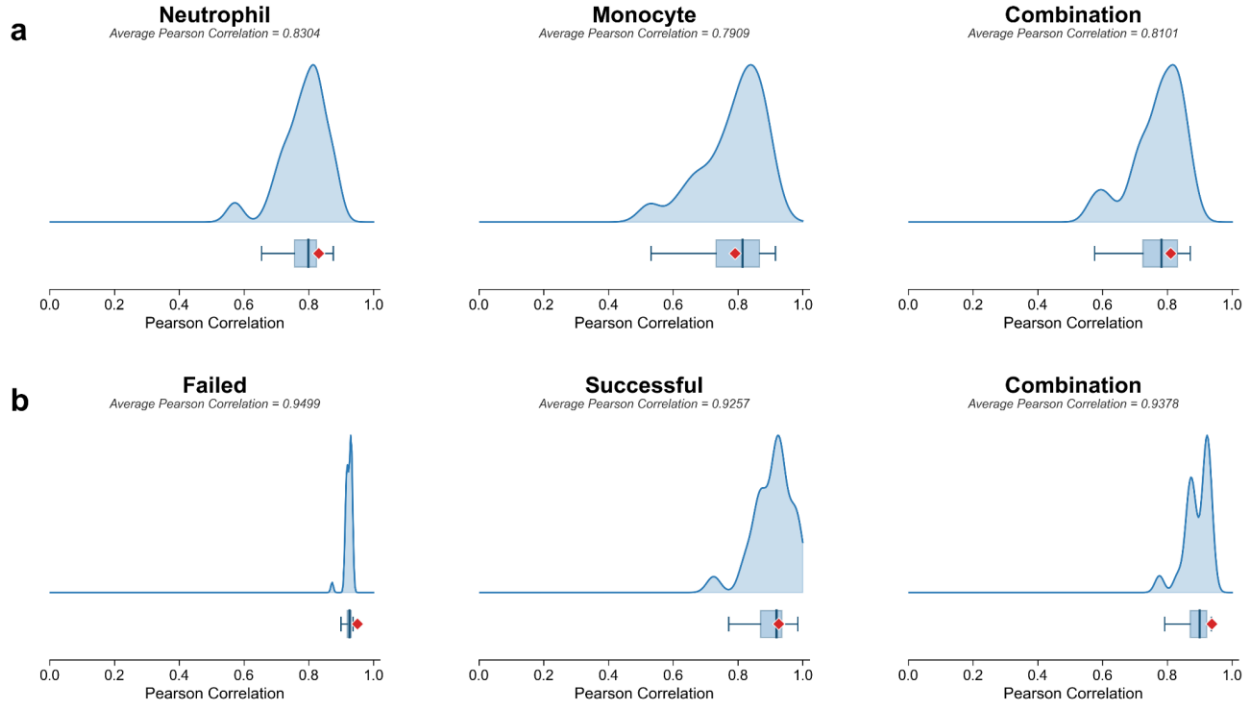


Figure S2. Correlation values show the predictive power of Deep Lineage in estimating gene expression profiles of unseen cells, related to Figures 3 and 5. The shaded density curve summarizes the distribution of Pearson correlations between predicted and observed gene expression profiles across clonal lineages. The box plot and red diamond indicate the interquartile range and mean, respectively. a, Correlation values for neutrophil, monocyte, and combined clonal lineages on day 6 of hematopoiesis. b, Correlation values for failed, successfully reprogrammed, and combined clonal lineages on day 28 of the reprogramming process. Datasets: mouse hematopoiesis (Weinreb et al.; GEO: GSE140802; a) and fibroblast-to-iEP reprogramming (Biddy et al.; GEO: GSE99915; b).

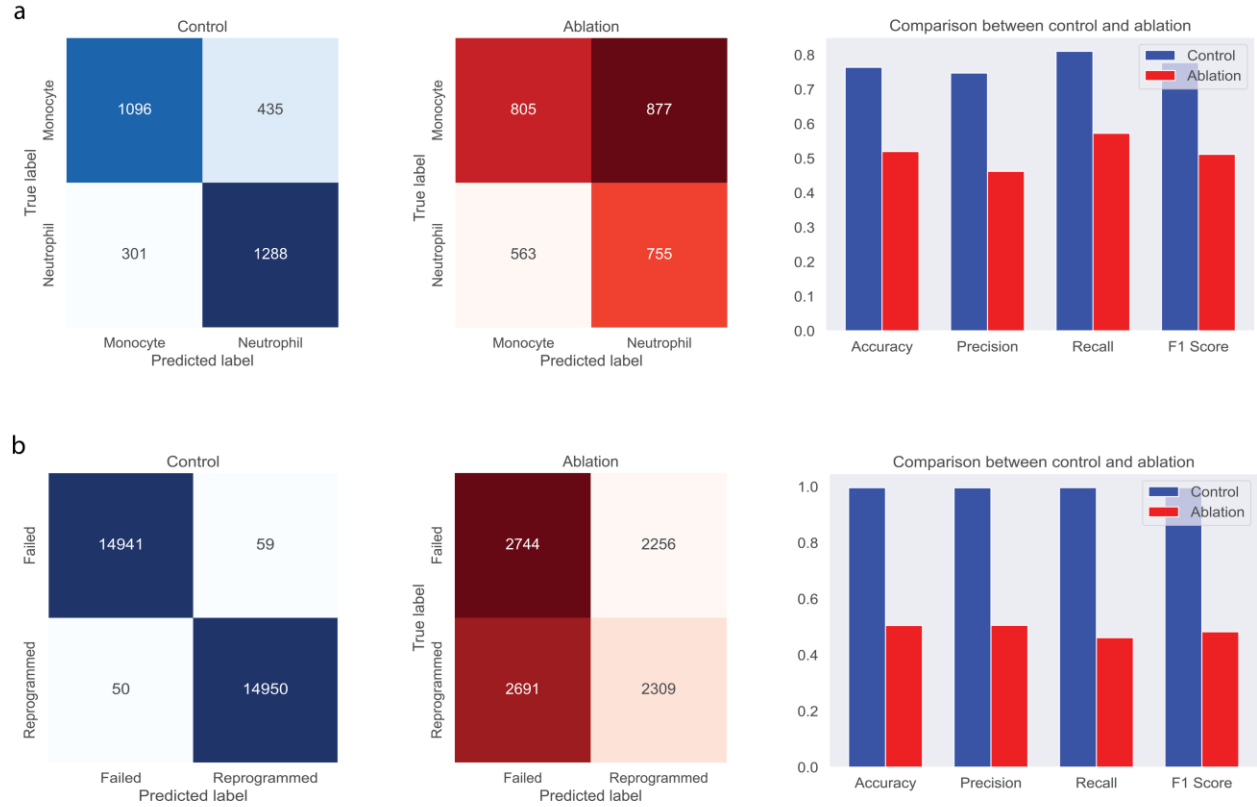


Figure S3. Ablation studies show that clonal structure is essential for accurate early cell fate prediction, related to Figures 4 and 6. To quantify the value of lineage information, ablation studies removed the clonal structure from hematopoiesis (a) and reprogramming (b) models. This forced the models to rely solely on transcriptomic profiles. Heatmaps and bar plots show classification metrics (accuracy, precision, recall, F1 score) between the full (Control) and ablated models. Removing clonal structure severely degrades performance across both datasets. Notably, reprogramming accuracy collapses to near-random chance. This demonstrates that early gene expression states alone are insufficient, and lineage history is essential for accurate early fate prediction. Datasets: mouse hematopoiesis (Weinreb et al.; GEO: GSE140802; a) and fibroblast-to-iEP reprogramming (Bidy et al.; GEO: GSE99915; b). Metrics (accuracy, precision, recall, F1) compare the full model (Control) with the clonal-structure-ablated model on held-out test clones; per-configuration values are listed in Tables S2 and S7. Ablated classification accuracy falls to ~0.52 (hematopoiesis) and ~0.51 (reprogramming), near the 0.50 chance level.

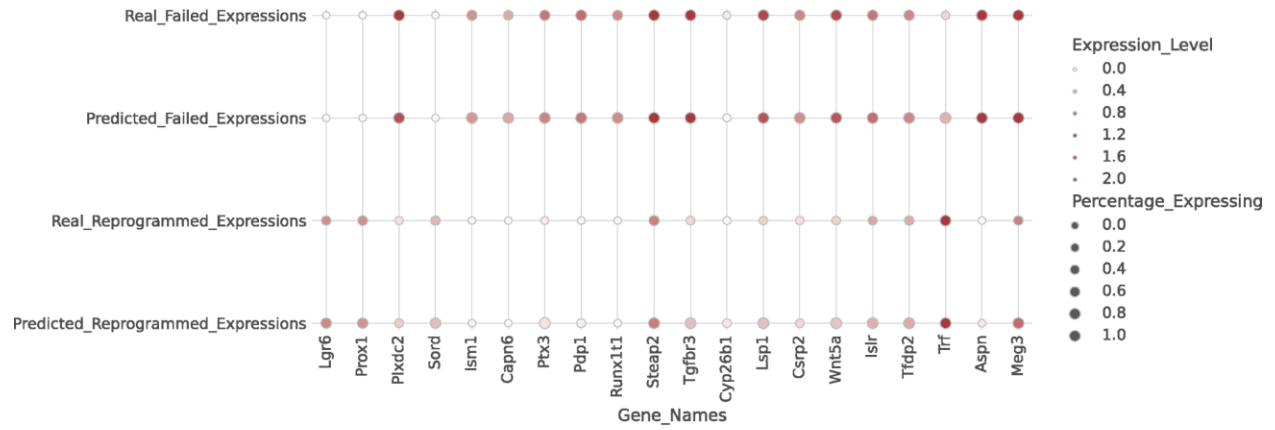


Figure S4. Expression prediction accuracy in fibroblast reprogramming, related to Figure 5. The dot plot illustrates the average expression of various genes between real cells and Deep Lineage predictions, for successful and failed reprogrammed cells. Dataset: fibroblast-to-iEP reprogramming (Biddy et al.; GEO: GSE99915). Each point is a gene; axes are mean predicted versus mean real expression for successful and failed reprogramming cells.

Hyperparameter	Values
Latent dimension	50, 75, 100, 150, 200
Hidden layer sizes	[512, 256] [1024, 512, 256] [2048, 1024, 512, 256] [4096, 2048, 1024, 512, 256]
Learning rate	1e-4, 5e-4, 1e-3, 2e-3
Batch size	1024, 2048, 4096

Table S1. Hyperparameter search space for autoencoder optimization, related to Figure 2 and STAR Methods. Hidden layer sizes define the encoder architecture (decoder mirrors the encoder). Total configurations searched: $5 \times 4 \times 4 \times 3 = 240$ configurations. The optimal configuration (latent dimension = 200, hidden layers = [1024, 512, 256], learning rate = 5e-4, batch size = 1024) was selected based on reconstruction accuracy on the validation set.

Gene	Roles	References
<i>Cryab</i>	Promotes osteogenic differentiation of human bone marrow stem cells via stabilizing β -catenin and promoting the Wnt signalling.	(1)
<i>Gpx3</i>	Involved in regulating the differentiation fate of Cardiac Fibroblasts.	(2)
<i>Col12a1</i>	Mediates cellular and extracellular mechanisms in development, regeneration, and disease.	(3)
<i>Il6st</i>	Plays a key role in cellular reprogramming, particularly in the generation of induced pluripotent stem cells.	(4)
<i>Gas1</i>	Modulates critical signaling pathways (Notch, SHH, Myc) making it a key player in cellular reprogramming and differentiation.	(5,6)
<i>Dlk1</i>	Regulates stem-cell pools and tissue differentiation by playing a pivotal role in controlling stem cell fate decisions and maintaining tissue homeostasis.	(7)
<i>Gata2</i>	GATA2 is crucial for maintaining hematopoietic stem cells (HSCs) and guiding their differentiation into various blood lineages. Mutations in GATA2 can result in cytopenias, bone marrow failure, and an increased risk of myeloid malignancies, known as GATA2 deficiency syndrome.	(8)
<i>Prtn3</i>	Prtn3 regulates myeloid differentiation by downregulating STAT3 expression, indicating a potential therapeutic target for the treatment of acute myeloid leukemia.	(9)
<i>Mmp8</i>	MMP8 plays a crucial role in fate and function in the hematopoietic niche. It facilitates HSC mobilization through the cleavage of extracellular matrix components and cell adhesion molecules in the bone marrow niche.	(10)
<i>Lpl</i>	Lpl regulates hematopoietic stem progenitor cell maintenance through DHA supply.	(11)

Table S8. Key genes identified in reprogramming and hematopoiesis with their associated functions, related to Figure 7. This table presents a curated list of top genes prioritized by Deep Lineage in both cellular reprogramming and hematopoiesis. Each row includes the gene symbol and a brief description of its known function or role, supported by relevant literature. The genes listed play diverse roles in stem cell maintenance, differentiation, signaling pathways, and tissue homeostasis.

Approach	Accuracy
x_0 expression $\rightarrow$ autoregressively predict x_2 type	0.785
$x_0 + x_1$ expression $\rightarrow$ predict x_2 type	0.891
x_2 expression $\rightarrow$ predict x_2 type (upper bound)	1.000

Table S11. Terminal-fate prediction accuracy of the autoregressive baseline on LineageSim data, related to STAR Methods. Terminal fate was predicted either directly from x_2 expression (upper bound), from x_2 expression forecast by the sequence-to-sequence LSTM given x_0 and x_1 , or fully autoregressively from x_0 alone (x_1 and then x_2 forecast before classification), followed by the feedforward cell-type classifier described in STAR Methods (Autoregressive formulation for cell fate prediction). Accuracies are computed on held-out LineageSim test trajectories (2,000 genes; 4,106 trajectories; 8,192 cells).

Supplemental references

1. Zhu, B., Xue, F., Li, G., and Zhang, C. (2020). CRYAB promotes osteogenic differentiation of human bone marrow stem cells via stabilizing β -catenin and promoting Wnt signalling. *Cell Prolif.* 53, e12709. <https://doi.org/10.1111/cpr.12709>.
2. Li, G., Qin, Y., Cheng, Z., Cheng, X., Wang, R., Luo, X., et al. (2022). Gpx3 and Egr1 are involved in regulating the differentiation fate of cardiac fibroblasts under pressure overload. *Oxid. Med. Cell. Longev.* 2022, 3235250. <https://doi.org/10.1155/2022/3235250>.
3. Izu, Y., and Birk, D.E. (2023). Collagen XII-mediated cellular and extracellular mechanisms in development, regeneration, and disease. *Front. Cell Dev. Biol.* 11, 1129000. <https://doi.org/10.3389/fcell.2023.1129000>.
4. Brady, J.J., Li, M., Suthram, S., Jiang, H., Wong, W.H., and Blau, H.M. (2013). Early role for IL-6 signalling during generation of induced pluripotent stem cells revealed by heterokaryon RNA-seq. *Nat. Cell Biol.* 15, 1244–1252. <https://doi.org/10.1038/ncb2835>.
5. Marczenke, M., Sunaga-Franze, D.Y., Popp, O., Althaus, I.W., Sauer, S., Mertins, P., et al. (2021). GAS1 is required for NOTCH-dependent facilitation of SHH signaling in the ventral forebrain neuroepithelium. *Development* 148, dev200080. <https://doi.org/10.1242/dev.200080>.
6. Lee, T.C., Li, L., Philipson, L., and Ziff, E.B. (1997). Myc represses transcription of the growth arrest gene gas1. *Proc. Natl. Acad. Sci. USA* 94, 12886–12891. <https://doi.org/10.1073/pnas.94.24.12886>.
7. Grassi, E.S., and Pietras, A. (2022). Emerging roles of DLK1 in the stem cell niche and cancer stemness. *J. Histochem. Cytochem.* 70, 17–28. <https://doi.org/10.1369/00221554211048951>.
8. Peters, I.J.A., de Pater, E., and Zhang, W. (2023). The role of GATA2 in adult hematopoiesis and cell fate determination. *Front. Cell Dev. Biol.* 11, 1250827. <https://doi.org/10.3389/fcell.2023.1250827>.
9. Liu, H., Sun, L., Zhao, H., Zhao, Z., Zhang, S., Jiang, S., et al. (2024). Proteinase 3 depletion attenuates leukemia by promoting myeloid differentiation. *Cell Death Differ.* 31, 697–710. <https://doi.org/10.1038/s41418-024-01288-4>.
10. Saw, S., Weiss, A., Khokha, R., and Waterhouse, P.D. (2019). Metalloproteases: on the watch in the hematopoietic niche. *Trends Immunol.* 40, 1053–1070. <https://doi.org/10.1016/j.it.2019.09.006>.
11. Liu, C., Han, T., Stachura, D.L., Wang, H., Vaisman, B.L., Kim, J., et al. (2018). Lipoprotein lipase regulates hematopoietic stem progenitor cell maintenance through DHA supply. *Nat. Commun.* 9, 1310. <https://doi.org/10.1038/s41467-018-03775-y>.